

**Untersuchungen zum Einsatz eines Standard-PC für die
Steuerung von dreiphasigen, gepulsten Stromrichtern**

Wolfgang Frank

DISTRIBUTION STATEMENT A
Approved for Public Release
Distribution Unlimited

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE 2000	3. REPORT TYPE AND DATES COVERED Dissertation	
4. TITLE AND SUBTITLE Untersuchungen zum Einsatz eines Standard-PC fuer die Steuerung von dreiphasigen, gepulsten Stromrichtern (Tests of the Use of a Standard PC for the Control of Three-Phase Pulse-Width Modulated (PWN) Converters)			5. FUNDING NUMBERS	
6. AUTHOR(S) Wolfgang Frank				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Fakultaet fuer Elektrotechnik (Energie- und Informationstechnik) Universitaet der Bundeswehr Muenchen				
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Universitaet fuer der Bundeswehr Muenchen Fakultaet fuer Wasserwesen Werner-Heisenberg-Weg 39 D-85577 Neubiberg GERMANY			10. SPONSORING/MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES Text in German, 144 pages.				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Category A; Public Release			12b. DISTRIBUTION CODE	
<p>ABSTRACT (Maximum 200 words)</p> <p>This dissertation explores the use of the standard PC as an alternative to the micro controller for controlling three-phase pulse-width modulated (PWM) converters. The PC-based approach has advantages in the areas of software, hardware, organization/management, and development potential. Taking into account that the micro controller-based approach requires a host PC, the costs associated with both options are comparable. One disadvantage of the PC-based approach is that almost all system-relevant functions (for example, interrupt management) are handled outside the processor and, therefore, cannot be as effectively integrated as they can be with highly integrated micro controllers</p> <p style="text-align: right; font-size: 2em; font-weight: bold;">20041008 584</p> <p>Machine assisted translation.</p>				
14. SUBJECT TERMS UNIBW, German, Personal Computer, Pulse-width modulated (PWM) converters, Software, Hardware, Processors, Interrupt management, Integrated micro controllers			15. NUMBER OF PAGES	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL	

UNIVERSITÄT DER BUNDESWEHR MÜNCHEN
Fakultät für Elektrotechnik
(Energie - und Informationstechnik)

Untersuchungen zum Einsatz eines Standard-PC für die Steuerung von dreiphasigen, gepulsten Stromrichtern

Dipl.-Ing. Wolfgang Frank

Vorsitzender des Promotionsausschusses:	Prof. Dr.-Ing. H. Bausch
1. Berichterstatter:	Prof. Dr.-Ing. L. Abraham
2. Berichterstatter:	Prof. Dr.-Ing. R. Marquardt

Tag der Prüfung: 10.01.2000

Mit der Promotion erlangter akademischer Grad:
Doktor-Ingenieur
(Dr.-Ing.)

Neubiberg, den 15.01.2000

AQ F04-12-1692

Vorwort

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter an der Professur für Leistungselektronik des Instituts für Elektrische Antriebstechnik der Universität der Bundeswehr München.

Herrn Prof. Dr.-Ing. L. Abraham danke ich sehr herzlich für seine Bereitschaft zum wissenschaftlichen Dialog, der sehr zum Gelingen dieser Arbeit beigetragen hat. Das verdient um so mehr meinen Respekt, als er dies auch nach seinem Ausscheiden aus dem aktiven Dienst an der Professur für Leistungselektronik aufrecht erhalten hat. In diesem Zusammenhang will ich ihm auch für die Übernahme des Referates danken.

Mein Dank gilt auch Herrn Prof. Dr.-Ing. R. Marquardt, der das Korreferat übernahm.

Der Fa. INDA Industrie- und Antriebstechnik GmbH, Ichenhausen, möchte ich für ihr Entgegenkommen bezüglich der Freistellung zur Vorbereitung auf meine Promotion danken.

Abschließend danke ich allen Mitarbeitern des Instituts, besonders Hr. Dr.-Ing. G. Bramm, Hr. Dr.-Ing. M. Reddig, Fr. M. Höcherl, Hr. K. Selzam, Hr. W. Eibl und Hr. J. Hofherr für die gute und überaus konstruktive Zusammenarbeit.

Publikationsliste

1. W. Frank: PWM-Inverter Drive Control Operating with a Standard Personal Computer, 2nd International Conference on Power Electronics & Drive Systems, Singapur, 26.-29.9.1997, Vol. 2, S. 774.
2. L. Abraham, W. Frank: Uncomplicated Inverter Drive for Research, Development and Teaching using Standard Personal Computer, 7th European Conference on Power Electronics and Applications, Trondheim, Norwegen, 8.-10.9.1997, Vol. 4, S. 4-952

Inhaltsverzeichnis

1 Einführung	1
1.1 Stand der Technik	1
1.2 Antriebssteuerung mit dem PC?	2
1.3 Zielsetzung und Überblick	4
2 PC-Architektur (AT-Modell)	6
2.1 Allgemeines Schema des AT-Modells	6
2.2 Beschreibung der Komponenten	9
2.2.1 Die CPU	10
2.2.1.1 Real Mode	10
2.2.1.2 Protected Mode	11
2.2.2 Der Interruptcontroller 8259A	12
2.2.3 Der Timer-Baustein 8254	13
2.2.4 Der DMA-Controller	14
2.2.5 Einsteckplätze (Slots)	15
2.2.6 Der digitale Ein-/Ausgabebaustein 8255	16
2.2.7 Tastaturcontroller (8042)	17
2.2.8 Speicher	18
2.2.8.1 Cache	18
2.2.8.2 Arbeitsspeicher (RAM)	19
2.3 Zusammenwirken der Bausteine im AT-System	19
2.3.1 Hardware-Programmierung	19
2.3.2 Interrupts	20
2.4 Betriebssysteme	22
2.4.1 Singletasking Systeme	23
2.4.2 Multitasking Systeme	23
2.4.3 Echtzeitsysteme	24
3 Antriebsprüfstand	25
3.1 Antriebsmaschine	25
3.2 Last	26
3.3 Antriebsstromrichter	26
3.4 PC für Steuerung und Regelung	28
3.4.1 Personal Computer (AT)	28

3.4.2 Zusatzkarten	29
3.4.2.1 Genauigkeit von Zeitfunktionen	29
3.4.2.2 Genauigkeit der Datenwandlung	30
3.4.3 Betriebssystem und Hochsprache	31
3.4.3.1 Betriebssystem	31
3.4.3.2 Programmiersprache	32
4 Grundlagen von Pulsverfahren	33
4.1 Gleichstromsteller	33
4.2 Phasenschalter	37
4.3 Dreiphasige Pulsverfahren	39
4.3.1 Stromrichterschaltung B6	39
4.3.2 Symmetrische dreiphasige Systeme	39
4.3.3 Nullkomponente der Augenblickswerte	40
4.3.4 Raumzeiger	41
4.3.4.1 Diagramm der Stromrichter-Spannungszustände	42
4.3.4.2 Stromraumzeiger	44
4.3.5 Modulationsgrad und Aussteuerungsgrad	45
4.3.6 Beschreibung verschiedener Pulsverfahren	46
4.3.6.1 Sinus-Modulation der Einzelphasen	47
4.3.6.2 Dreiphasiges Pulsen mit symmetrischen Nullzuständen	48
4.3.6.3 Zweiphasiges Pulsen erster Art	50
4.3.6.4 Zweiphasiges Pulsen zweiter Art	52
5 Stromrichtersteuerung	53
5.1 Interruptsteuerung	54
5.1.1 Multizähler-Verfahren	54
5.1.2 Monozähler-Verfahren	55
5.2 Steuereingriffe durch Interrupts	56
5.2.1 Berechnungsalgorithmus	56
5.2.2 Interruptroutine	59
5.3 Spezifische Zusatzroutinen	60
5.3.1 Phasenumschaltung	60
5.3.2 Ende eines Ausführungszeitraums	60
5.3.3 Meßdatenerfassung	60
5.4 Sonderfälle	61
5.4.1 Verzögerungen durch die Laufzeit von "outp"-Befehlen	61
5.4.2 Verzögerungen durch Interrupts	62

5.5 Maßnahmen für Sonderfälle	65
5.5.1 Maßnahmen erster Art	65
5.5.2 Maßnahmen zweiter Art	66
5.5.2.1 Vereinigungsmethode	67
5.5.2.2 Warteschleifenmethode	69
5.5.2.3 Vergleich beider Verfahren	74
5.5.3 Umschaltunterdrückung	75
5.5.3.1 Umschaltunterdrückung am Anfang einer Halbperiode ..	77
5.5.3.2 Umschaltunterdrückung am Ende einer Halbperiode	77
5.5.3.3 Mehrfachunterdrückungen	78
5.5.3.4 Programmtechnische Realisierung	79
5.5.4 Ingangsetzen und Abschalten des Stromrichters	81
5.6 Umsetzung der Pulsverfahren	82
5.6.1 Sollwertraumzeiger und Berechnungstabelle	82
5.6.2 Verfahren mit Nullkomponente	83
5.6.3 Einphasige Pulsverfahren	85
5.7 Regelung	86
5.7.1 Meßwerte	87
5.7.2 Stromregelung	89
5.7.3 Überlagerte Regelung	89
5.7.4 Zeitfenster	90
5.7.4.1 Berechnung nach Steuereingriff	91
5.7.4.2 Berechnung durch Auslesen des Zeitwerks	92
5.7.4.3 Berechnung durch Laufzeitmessung	92
5.7.5 Begrenzung des Sollwert-Spannungsraumzeigers	93
6 Kommunikation	96
6.1 Tastatur	96
6.2 Multi-I/O-Karten	98
6.3 Bildschirm	99
6.3.1 On-line-Ausgabe	99
6.3.2 Off-line-Ausgabe	100
6.4 Interrupts	101
6.5 Schnittstellen	101
6.6 Netzwerkkarten	102
7 Schutzeinrichtungen	103
7.1 NOT-AUS	103

7.2 Halbleiterschutz	103
7.2.1 Direkter Schutz mit Interrupt	103
7.2.2 Indirekter Schutz durch Softwareabschaltung	104
8 Experimentelle Untersuchungen	105
8.1 Meßaufbau und Meßtechnik	105
8.1.1 Meßgeräte	106
8.1.2 Spannungsmessung	107
8.1.3 Strommessung	107
8.2 Messungen zur Halbperiodensteuerung	108
8.3 Messungen zu den Sonderfällen der Stromrichtersteuerung	110
8.3.1 Warteschleife	110
8.3.2 Umschaltunterdrückung	111
8.4 Betrieb des Asynchronmotors	113
8.4.1 Betrieb mit sinusförmigem Verlauf des Spannungs-Sollwert- Raumzeigers	115
8.4.2 Betrieb bei kleinen Modulationsgraden ($M \approx 0.3$)	115
8.4.3 Betrieb bei Begrenzung	119
8.4.3.1 Passive Begrenzung	119
8.4.3.2 Aktive Begrenzung	121
9 Zusammenfassung und Ausblick	123
10 Literaturverzeichnis	126
11 Verzeichnis der Abkürzungen und Indizes	134
A PC- und Prozessorinterna	136
A.1 Registersatz	136
A.2 Belegung der Interruptleitungen	137
A.3 Interrupt-Descriptor-Table (IDT)	138
A.4 Scancode der Tastatur	138
B Hardware-Interruptroutinen	141
C Zeitmeß-System mit geringem Meßfehler	143

1 Einführung

1.1 Stand der Technik

In der Antriebstechnik haben sich für Steuerungen und Regelungen heute Lösungen mit Mikrocontrollern oder DSP entsprechend dem grau schattierten Teil in Bild 1.1 durchgesetzt. Sie enthalten neben der Mikrocontrollereinheit, dem Stromrichter und dessen Versorgungsgerät meist sogar auch die Antriebsmaschine selbst. Je nach Komplexität des Antriebs wird aus der vergleichsweise großen Zahl an Mikrocontrollern der geeignete Typ ausgesucht und eingesetzt. Für die Entwicklung und Prüfung solcher Antriebe ist aber ein zusätzlicher, überlagerter Teil notwendig, der heute neben den Maschinen meist noch einen PC enthält (unschattierter Teil in Bild 1.1). Die Anforderungen an letzteren sind aufgrund der komplexen Software-Entwicklungsumgebung, mit der die Software für den Antrieb erstellt wird, sehr hoch und er muß daher sehr leistungsfähig sein. Die Entwicklungsumgebung ist genau auf den benutzten Controller zugeschnitten und kann nur eingeschränkt für andere Controller benutzt werden. Die Kosten für solche Umgebungen liegen z.B. für DSP im Bereich mehrerer tausend Mark. Die Programmierung erfolgt aus Gründen der Rechenleistung oft in Assembler. Über eine Schnittstelle des Rechners (z.B. seriell oder parallel) wird der Binärcode in den Speicher des Mikrocontrollers geladen. Er steuert und regelt den Wechselrichter für die Antriebsmaschine, wobei er ggf. auch Steuerungsaufgaben für dessen Stromversorgung übernehmen kann. Die Antriebsmaschine ist an eine Lastmaschine gekoppelt, die über ein eigenes Steuergerät verfügt. Da der PC während des Mikrocontrollerbetriebs zur Verfügung steht, ist über ihn die Vorgabe von Sollwerten für das Steuergerät der Last möglich.

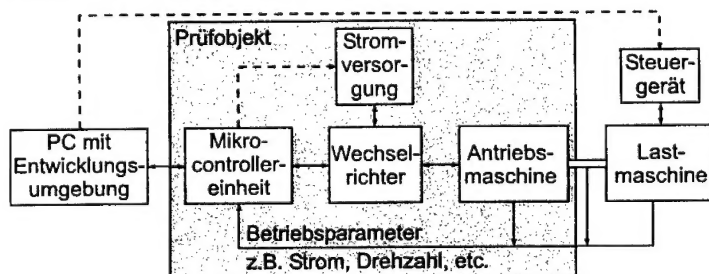


Bild 1.1: Schema eines modernen Antriebsprüfstandes mit Mikrocontroller

Um dennoch die Entwicklungszeiten von Antrieben, die auf dem beschriebenen System basieren, kurz zu halten, muß das Wissen um die Programmierung der Mikrocontroller in den Ent-

wicklungsabteilungen sofort abrufbar sein. Dies ist jedoch nur mit einem erhöhten Personalaufwand möglich, da meist allein für die Programmierung der Mikrocontroller eine oder sogar mehrere Arbeitskräfte zuständig sind.

1.2 Antriebssteuerung mit dem PC?

Etwa seit 1990 hat eine enorme Entwicklung der Rechen- bzw. der Systemleistung von PC eingesetzt. Bild 1.2 zeigt angenähert den Verlauf der Rechenleistung von Intel-Prozessoren für die Jahre 1972 bis 2000. Gründe der stark angestiegenen Leistungsfähigkeit sind der Übergang zu immer kleineren Transistorstrukturen vor allem bei der CPU und dem Speicher, die immer höhere Taktraten zulassen, sowie neuartige Busse (z.B. PCI-Bus) und Schnittstellen (z.B. USB-Schnittstelle).

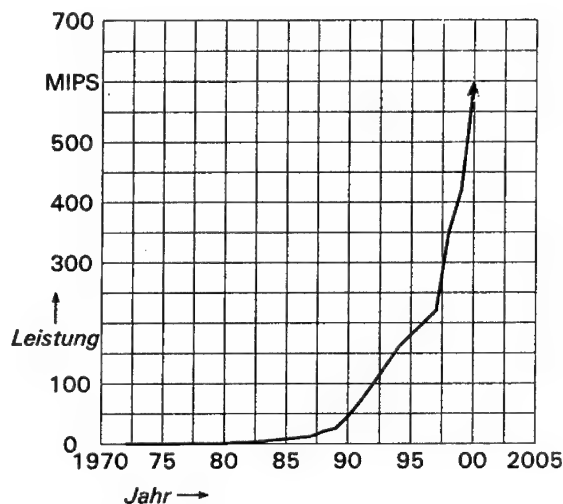


Bild 1.2: Entwicklung der Rechenleistung von Intel-Prozessoren von 1972-2000 (geschätzt) in Millionen Instruktionen pro Sekunde (MIPS)

Dadurch eröffnen sich dem PC neben dem traditionellen Gebiet der Textverarbeitung auch neue Einsatzmöglichkeiten in den Bereichen Multimedia, Wissenschaft und Technik. Studien (z.B. [1.1]) belegen, daß gerade für Steuerungen auf PC-Basis in den kommenden Jahren sehr hohe Zuwachsraten (1998-2000: 535%) zu erwarten sind. Vor diesem Hintergrund stellte sich die Frage, ob es sinnvoll ist, den Host-PC von Bild 1.1 selbst als Steuergerät für Stromrichterantriebe zum Einsatz in Forschung, Entwicklung und Lehrbetrieb heranzuziehen.

Bild 1.3 zeigt die Vorteile eines solchen Systems, die sich in vier Kategorien aufteilen lassen:

- **Hardware:** PC verfügen über vergleichsweise viel Arbeitsspeicher und enthalten eine weithin bekannte Hardware. Über Einsteckkarten lassen sich fehlende Funktionen (z.B. Analog-Digital-Wandlung, etc.) einfach und ohne zusätzliche Schaltungsentwicklung

nachrüsten. Die leistungsfähige CPU ist mit allen Vorgängermodellen abwärtskompatibel.

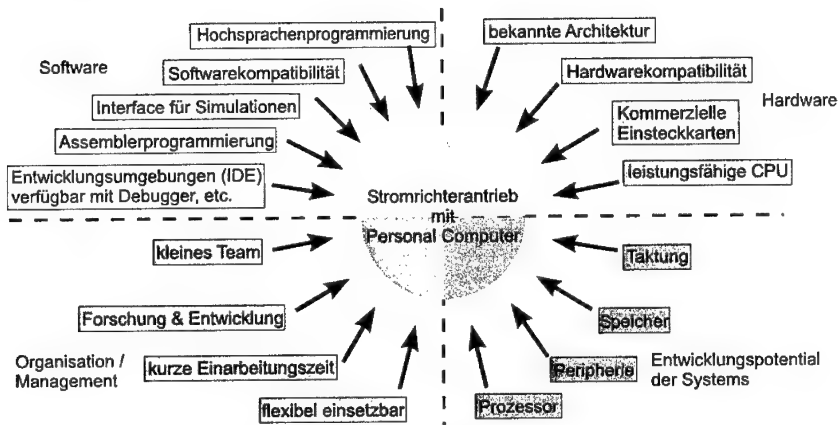


Bild 1.3: Vorteile eines Stromrichterantriebs mit einem PC als Steuergerät

- **Software:** Für PC existieren eine große Anzahl an Entwicklungsumgebungen für verschiedene Hochsprachen (z.B. C, Pascal, Modula2 usw.), welche die Softwareentwicklung mit diversen Funktionen, wie integrierte Debugger, unterstützen. Trotzdem können solche Systeme nach wie vor in Assembler programmiert werden. Vorhandene Software bleibt auch nach dem Übergang zu schnelleren Konfigurationen aufgrund der oben erwähnten Kompatibilität funktionsfähig. Die für die Steuerung relevanten Programmteile können im Original auch für die Simulation des Antriebssystems benutzt werden. Eine zusätzliche Modellierung der Steuerung für spezielle Simulationsprogramme entfällt somit.
- **Organisation / Management:** Durch die Vorteile auf der Hardware-Seite ist nur geringes zusätzliches Fachwissen über die Hardware von PC erforderlich. Dies darf heute vielfach schon als bekannt vorausgesetzt werden, was die Einarbeitungszeit entscheidend verkürzt. Deshalb sind diese Systeme gerade für den Forschungs- und Entwicklungsbereich mit den meist sehr kleinen Arbeitsgruppen sinnvoll. Durch die Variation von Zusatzkomponenten, wie Einsteckkarten, sind PC flexibel einsetzbar.
- **Entwicklungspotentiale:** Nicht zuletzt ist auch die Aussicht auf steigende Systemleistung bei weiterhin beibehaltener Kompatibilität zu den Vorgängersystemen ein Grund solche Systeme einzusetzen. So können z.B. nachträglich notwendig gewordene Funktionen durch Übergang auf eine leistungsfähigere CPU eingefügt werden. Daneben

bieten auch die Peripherie oder der Arbeitsspeicher noch zusätzliche Entwicklungspotentiale.

Allerdings muß festgestellt werden, daß im Vergleich zu Mikrocontroller-Systemen, deren Hardware meist nur wenige 10-100 DM kostet, die Vorteile von PC-basierenden Steuerungssystemen durch weitaus höhere Hardware-Kosten erkaufte werden müssen. Diese Tatsache relativiert sich allerdings bei Betrachtung der gesamten Kosten für das jeweilige Entwicklungssystem. Im Fall eines Mikrocontrollersystems kommen nämlich zusätzlich die Aufwendungen für den Host-PC sowie die hohen Kosten für die integrierte Entwicklungsumgebung hinzu. In der Summe sind die Kosten für ein solches System also meist genau so hoch oder sogar höher als für ein PC-basierendes System. Dessen Gesamtkosten liegen derzeit für die Hardware (PC und Einsteckkarten) allein bei ca. DM 4000-6000. Die notwendige Software (z.B. Hochsprachencompiler) ist im Gegensatz zu Mikrocontrollersystemen jedoch vergleichsweise billig oder kann sogar kostenlos bezogen werden.

1.3 Zielsetzung und Überblick

In der Vergangenheit gab es bereits mehrfach Ansätze, Umrichter oder Maschinen mit PC bzw. Mikroprozessoren zu steuern. Einige davon berechnen die Pulsmuster off-line, speichern diese Daten in Tabellen ab und geben sie während des Betriebs nacheinander aus ([1.2], [1.3], [1.4]). Weitere Möglichkeiten bestehen in der Benutzung vereinfachter Modulationsarten oder durch Programmierung in Maschinensprache ([1.5], [1.6]).

Diese Arbeit soll Aufschluß darüber bringen, inwiefern PC geeignet sind, als Echtzeit-Steuergerät für statische Umrichter von Antriebssystemen herangezogen zu werden. Es werden hier Pulsfrequenzen bis zu 5 kHz angestrebt. Dazu waren Untersuchungen auf folgenden Gebieten notwendig:

- **Programmierung:** Anhand einfacher Messungen soll zunächst abgewogen werden, welches der verfügbaren, populären Betriebssysteme für diese Aufgabe am besten geeignet ist. Ein Kriterium dafür ist beispielsweise das Zeitverhalten des Systems bei Interrupts. Die Programmierung in Assembler ergibt meist die kürzesten Laufzeiten. Hochsprachen bieten dagegen vielfältige standardisierte Funktionen. Es ist deshalb zu untersuchen, inwieweit nicht große Teile der Software in Hochsprachen erstellt werden können, die den Echtzeitbetrieb aber trotzdem gewährleisten. Damit wird der Entwicklung Rechnung getragen, daß die Programmierung in Assembler vor allem in den letzten Jahren immer mehr von modernen Hochsprachen verdrängt wird.

Ferner stellt sich die Frage, mit welchen Softwarestrukturen ein beliebiger Wechsel von Pulsverfahren - selbst während des Betriebs - am besten realisiert werden kann. Es ist deswegen eine größtmögliche Flexibilität der Betriebsparameter (z.B. Aussteuerung, Pulsfrequenz, usw.) vorzusehen. Die Pulsverfahren sollen außerdem in Echtzeit berechnet und ausgegeben werden. Unklar ist auch die Konfiguration von Übergabeparametern bei modularem Aufbau der Software, sodaß einzelne Prozeduren auch z.B. für Simulationen benutzt werden können

- **Hardware:** Eigene Entwicklungen sind in diesem Bereich nicht vorgesehen. Ein kommerzieller Leistungsteil - bestehend aus einer ungesteuerten B6-Brücke als Gleichrichter, einem Spannungszwischenkreis und einer gesteuerten B6-Brücke (3 IGBT-Halbbrücken mit Ansteuermodulen) als Wechselrichter - wird von einer handelsüblichen PC-Ein-/Ausgabekarte angesteuert. Das erfordert aber dennoch, daß die einzelnen Bestandteile auf ihr Betriebsverhalten hin getestet werden, da Parameter, wie z.B. die Laufzeiten der Gatesignale im Ansteuermodul, durchaus Auswirkungen auf die Steuerung haben können.
- **Steuerung / Regelung:** In dieser Arbeit wird die Steuerung und Regelung einer dreiphasigen Asynchronmaschine angestrebt. Dieser Motortyp besitzt eine ausreichende Komplexität, um auch Abschätzungen über die Grenzen eines solchen Systems geben zu können. Die Ausgabe der Steuersignale für den Pulswechselrichter soll aus Gründen der Rechenleistungsoptimierung auf der Grundlage von Interrupts erfolgen. Diese stellen wegen des "funktional verteilten" Systems eines PC¹ einen wichtigen Parameter hinsichtlich der Grenzen der Systemleistung dar. Deshalb soll zunächst der Zeitbedarf von Interrupts untersucht werden. Daraus können sich Erkenntnisse darüber ergeben, ob bzw. durch welche speziellen Eingriffe in die Hard- und Software ggf. Verzögerungen im Programmablauf umgangen oder kompensiert werden können, sodaß sich möglicherweise höhere Pulsfrequenzen realisieren lassen.

Die oben genannten Kriterien beschreiben damit einen Aufbau (siehe Abschnitt 3), der die Möglichkeiten eines Systems auf Mikrocontroller-Basis mit den vielfältigen und flexiblen Eigenschaften eines PC in geeigneter Weise für den Betrieb von stromrichtergesteuerten Maschinen miteinander verbindet.

¹Darunter ist zu verstehen, daß in PC fast alle systemrelevanten Funktionen (z.B. Interruptverwaltung) außerhalb des Prozessors liegen und daher nicht so effektiv miteinander verbunden werden können, wie das bei hoch integrierten Mikrocontrollern der Fall ist.

2 PC-Architektur (AT-Modell)

2.1 Allgemeines Schema des AT-Modells

Der Begriff "PC" bezeichnet eigentlich ein Rechnersystem, das einem bestimmten Anwender zugeordnet ist. Mit der Entwicklung verschiedener Prozessoren entstanden unterschiedliche PC-Systeme. In dieser Arbeit werden jedoch nur Systeme auf der Grundlage von Intelprozessoren behandelt, da diese in Europa am weitesten verbreitet sind¹. Solche PC sind durch das Advanced Technology (AT) Modell spezifiziert. Das AT-Modell ist für Systeme gültig, die einen 80286- oder jüngeren Prozessor beinhalten. Es besagt nicht nur, daß bestimmte Funktionen implementiert sein müssen, es sorgt auch für die einheitliche Anbindung von Komponenten, wie z.B. des Timerbausteins, über standardisierte Geräteadressen. Ältere XT-Rechner besitzen einen 8086- oder 8088-Prozessor. Diese werden jedoch nicht in dieser Arbeit behandelt.

Die PC gehören zur Gruppe der "Von-Neumann-Maschinen" mit "Princeton-Architektur", die einen gemeinsamen Speicher für Daten und Befehle besitzen [2.1]. Deshalb werden sowohl Daten wie Befehle auf denselben Busleitungen transportiert. Das hat zur Folge, daß zu einem Zeitpunkt nur Daten oder nur Befehle übertragen werden können. Die zuvor übermittelte Adresse bestimmt, ob die übertragene Information einen Befehl oder ein Datum darstellt. Systemerweiterungen (z.B. mit Einsteckkarten) können damit sehr einfach durchgeführt werden, indem den Erweiterungen bestimmte Speicherbereiche innerhalb des sogenannten "I/O-Adreßraums" zugeordnet werden. Allerdings kann dies zu Zugriffskonflikten auf dem Bus führen, was eine zusätzliche Busverwaltung erfordert.

Eine zweite Gruppe bilden Systeme, deren Prozessoren auf der sogenannten "Harvard-Architektur" aufbauen (siehe Literatur [2.2]). Bei diesen Rechenanlagen ist der Datenspeicher vom Befehlsspeicher räumlich getrennt, so daß über eigene Bussysteme Daten und Befehle gleichzeitig übertragen werden können. Dies führt zu einer erheblichen Rechenleistung. Systemerweiterungen sind jedoch vergleichsweise schwierig zu realisieren weshalb solche Rechenanlagen nur begrenzt einsetzbar sind.

¹Weitere PC-Familien sind Macintosh-PC und PowerPC, die mit einer Motorola M680x0-CPU ausgestattet sind oder Rechner mit Alpha-CPU von Digital Equipment.

Gemeinsam mit anderen PC (z.B. Motorola-Systeme) zeigt sich das AT-Modell als "offenes System", da solche Rechner nicht auf eine feste Anzahl von Funktionen begrenzt sind, sondern sich beinahe beliebig verändern und erweitern lassen. Dies wird durch die Einbindung von standardisierten Bussystemen, wie z.B. des ISA- oder des PCI-Busses, und Schnittstellen (seriell, parallel, USB¹) erreicht.

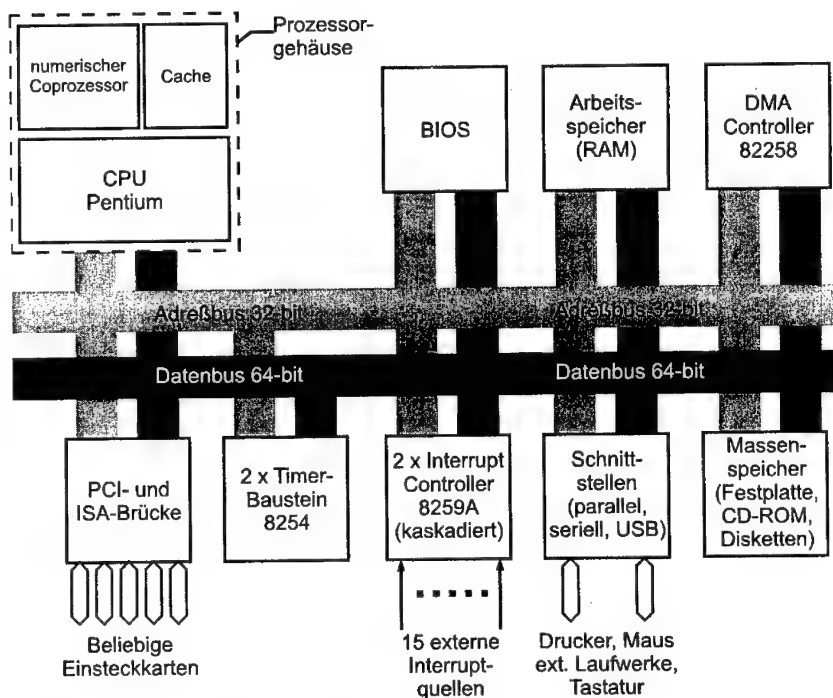


Bild 2.1: Schema des AT-Modells (Pentium-System)

Bild 2.1 zeigt den prinzipiellen Aufbau eines AT-Systems. Im Kern steht die CPU mit Intel-Architektur [2.3], die über den Systembus - bestehend aus dem Daten- und Adreßbus - mit dem Basic Input Output System (BIOS) und dem Arbeitsspeicher verbunden ist. Im BIOS-Baustein sind alle zur grundsätzlichen Funktion des PC notwendigen Daten abgespeichert, z.B. die Anfangsadressen der Initialisierungsroutinen sowie die Boot-Sequenz für den Start des Systems und die Initialisierungsdaten der Hardwarekomponenten. Das Betriebssystem

¹USB (= Universal Serial Bus) ist ein im Jahre 1996 definiertes Bussystem [2.4]

paßt diese Voreinstellungen noch an und stellt Erweiterungen zur Verfügung. Das betrifft insbesondere die Verarbeitung von Ausnahmefällen (Exceptions) und Systeminterrupts.

Abhängig vom verwendeten AT-System besitzt der Datenbus eine Breite von 16-bit (286), 32-bit (386, 486) oder 64-bit (Pentium). Bei der Steuerung und Regelung von Stromrichtern fallen im Normalbetrieb nur geringe Datenmengen an, sodaß ein Adreß- und Datenbus mit einer Breite von je 16 Bit eigentlich ausreichen würde. Aus Gründen der Rechenleistung ist jedoch eine hohe Taktrate der CPU unerlässlich (möglichst 400 MHz). Systeme mit derartigen Prozessoren bieten aber stets breitere Busse (32-bit Adreßbus, 64-bit Datenbus) an. Dies hat den Vorteil, daß der Rechner, bei Ausrüstung mit dem entsprechenden Speicher, über eine genügend hohe Rechenleistung verfügt, um auch (Off-line-) Simulationen des Stromrichter-systems mit der Originalsoftware für die Steuerung und Regelung zu bewältigen. Je nach Ausführung des Prozessors sind auch Teile außerhalb der CPU in den Prozessor integriert, wie z.B. der numerische Koprozessor.

Alle außerhalb des Prozessors und des Prozessorbusses gelegenen Bausteine werden im folgenden als "extern" bezeichnet. In Bild 2.1 sind dies der Arbeitsspeicher (RAM) mit dem Baustein für direkten Speicherzugriff (DMA-Controller), der Timerbaustein (8254), der Interrupt-controller (8259A), der Baustein zur Verwaltung des ISA- oder PCI-Busses (s. Bild 2.1 "PCI- und ISA-Brücke") sowie die Schnittstellen-Bausteine für die serielle, parallele und USB-Schnittstelle. Einige dieser Komponenten sind im sogenannten "Chip-Set" integriert und somit nicht mehr als getrennte Bausteine erkennbar, wenngleich die Art der Programmierung und deren Funktion erhalten bleibt. Der ISA- und der PCI-Bus, die mittels der ISA- bzw. PCI-Brücke mit dem Systembus verbunden sind, stellen Einsteckplätze für Zusatzkomponenten bereit und dienen so der oben erwähnten Fähigkeit zur Veränderung und Erweiterung des Systems.

Ein weiteres Merkmal von PC ist die notwendige Existenz eines Betriebssystems (z.B. DOS, Windows, Linux). Es muß zwar auf die beschriebene Konfiguration abgestimmt sein, ist aber nicht direkter Bestandteil des AT-Modells. Es kann damit ohne Änderung der Hardware bei jedem neuen Startvorgang gewechselt werden. Die unterschiedlichen Arten von Betriebssystemen sind in Abschnitt 2.4. beschrieben.

2.2 Beschreibung der Komponenten

Für die Anwendung bei der Steuerung und Regelung des Stromrichters sind einige Komponenten aus Bild 2.1 von untergeordneter Bedeutung. Das sind:

- Massenspeicher, wie z.B. Festplatten, CD-ROM oder Diskettenlaufwerke: Die Zugriffszeit solcher Geräte liegt im Bereich mehrerer Millisekunden und ist daher so lang, daß sie im On-line-Betrieb nicht benutzt werden dürfen. Eine Ausnahme wäre nur dann denkbar, wenn sich in der Steuerung und Regelung eine Mindestzeitspanne ohne Regeleingriff ergeben würde, die einen solchen Speicherzugriff erlaubt. Das ist aber bei der angestrebten Pulsfrequenz bisher nicht vorgesehen.
- Schnittstellen (seriell, parallel, USB): Wenngleich sie einen ggf. höheren Datendurchsatz aufweisen, gelten im Prinzip die gleichen Aussagen wie für die Massenspeicher. Da die Schnittstellen ebenfalls interruptgesteuert arbeiten, kann es mit den Interrupts für die eigentliche Stromrichtersteuerung zu Konflikten kommen, deren Vermeidung zu komplexen Programmstrukturen führt.
Die Steuerung von Stromrichtern über die vorhandenen Schnittstellen wäre denkbar, allerdings erfordert das einen zusätzlichen Schaltungsaufwand (z.B. Spannungsversorgungen oder Interfaceschaltungen). Außerdem bieten die Schnittstellen eine zu geringe Anzahl an frei verfügbaren Ein-/Ausgabeleitungen, so daß eine zusätzliche Steckkarte dennoch notwendig wäre.
- Coprozessor und Cache: Beide Prozessorteile dienen zur Beschleunigung des Systems, was für die Anwendung äußerst wichtig ist. Da aber der Coprozessor und der Cache direkt von der CPU angesteuert werden, sind deren innerer Aufbau für die Programmierung in einer Hochsprache ohne Bedeutung.

In den folgenden Unterabschnitten wird die Funktion der anderen Komponenten beschrieben, soweit sie für die vorgesehene Anwendung wichtig sind. Dabei wird zunächst auf die Arbeitsweise dieser Komponenten unter "normalen" Betriebsbedingungen eingegangen. Darunter ist zu verstehen, daß auf dem PC ein Betriebssystem installiert ist und der PC ausschließlich kommerzielle Anwendungen (z.B. EDV-Anwendungen, Spiele, etc.) ausführt. Sobald vom Nutzer eines PC selbst programmierte Routinen zum Einsatz kommen, welche die Komponenten eines PC manipulieren, wird in dieser Arbeit von "besonderen Betriebsbedingungen" gesprochen.

2.2.1 Die CPU

Die CPU [2.5] ist natürlich das Kernstück einer digitalen Rechneranlage, da sie entscheidend die Rechenleistung des Systems bestimmt. Die internen Prozessorstrukturen beeinflussen allerdings weniger die in dieser Arbeit angestrebte Hochsprachenprogrammierung als vielmehr den Compilercode. Es wird daher hier nicht darauf eingegangen. Es ist aber wichtig, die beiden Betriebsarten "Real Mode" und "Protected Mode" für die Speicherverwaltung von 80x86-Prozessoren näher zu beschreiben, um entscheiden zu können, welche für die vorgesehene Aufgabe besser geeignet ist.

Die Gesamtheit des maximal adressierbaren Speichers wird "physikalischer Adreßraum" [2.6] genannt und ist durch die Breite des Adreßbusses begrenzt. Natürlich steht in der Praxis nicht hinter jeder physikalischen Adresse eine Speicherzelle, da aus Kostengründen der reale Arbeitsspeicher häufig nur einen kleinen Teil des physikalischen Adreßraums ausfüllt. Der physikalische Adreßraum ist von Null an fortlaufend numeriert. Programme und Anwendungen greifen darauf allerdings nicht direkt zu. Sogenannte "Speicherkonzepte" bilden den physikalischen Adreßraum auf den "logischen Adreßraum" ab, der den Anwendungen zur Verfügung steht. Zusätzlich existiert noch der sogenannte "I/O-Adreßraum" [2.7]. Über diese Adressen werden die Register von externen Bausteinen, wie z.B. dem Systemtimer, programmiert. Der I/O-Adreßraum wird vom physikalischen Adreßraum mit Hilfe des Steuersignals M/I/O# der CPU unterschieden.

2.2.1.1 Real Mode

Beim Real Mode bildet der Prozessor die physikalische Adresse aus dem Inhalt zweier Prozessorregister mit je 16 Bit. Das eine Register (Segmentregister) stellt den Beginn eines sogenannten Speichersegmentes dar. Mit diesem Register können 2^{16} unterschiedliche Segmente vereinbart werden. Das andere Register (IP-Register) gibt den sogenannten "Offset" an und legt die Lage der Speicherzelle innerhalb eines Segmentes fest. Für die Ermittlung der aktuel-

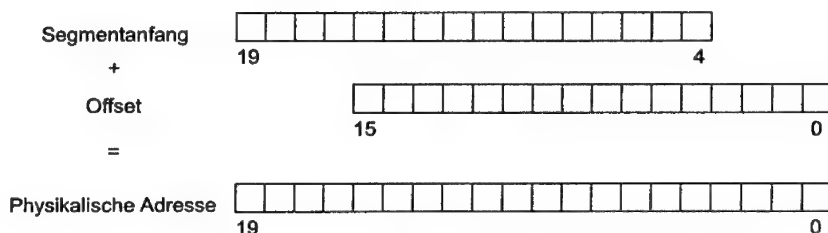


Bild 2.2: Berechnungsvorschrift zur Ermittlung der physikalischen Adresse aus den Angaben des Segments und des Offsets

len Speicheradresse multipliziert der Prozessor zunächst das Segmentregister mit 16 und addiert dann den Offset (siehe Bild 2.2). Solange man im gleichen Segment bleibt, muß nur der Offset übertragen werden. Dadurch erreichte man bei XT-Systemen mit 16-bit breitem Bus eine Erhöhung der Systemleistung. Die Startadresse der Segmente kann in einem Raster von $2^4=16$ Byte im physikalischen Adreßraum festgelegt werden (siehe Bild 2.3). Dieser Algorithmus bewirkt, daß nicht $2^{16} \times 2^{16}$ Speicherzellen (=Anzahl der Segmente * Anzahl der Speicherzellen eines Segments) adressierbar sind, sondern nur auf 16×2^{16} Byte = 1 MB zugegriffen werden kann, obwohl 2^{16} unterschiedliche Segmente vereinbart werden können. Dies hat den Vorteil, daß der physikalische Adreßraum sehr dicht beschrieben werden kann. Zwei Programmteile können so mit einem minimalen Abstand von 16 Byte im Arbeitsspei-

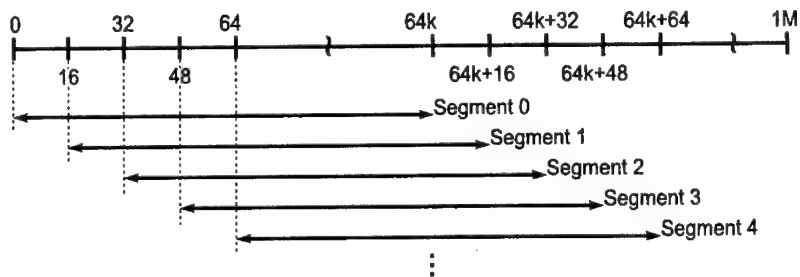


Bild 2.3: Lage der einzelnen Segmente im physikalischen Adreßraum

cher abgelegt werden (Bild 2.3). Für Systeme mit nur 16-bit breiten Bussen ist dies eine effektive Art, den Speicher dicht beschreiben zu können. Im Falle von Segmentgrenzen, die nur bei Vielfachen von 64 kB lägen, würden sich ggf. größere Speicherbereiche ergeben, die nicht beschrieben würden, wenn eine Anwendung nicht das volle Segment belegen würde und der Speicher wäre damit schlecht genutzt. Moderne PC verfügen über weit mehr als 1 MB Arbeitsspeicher, wovon im Real Mode trotzdem nur 1 MB adressiert werden kann.

2.2.1.2 Protected Mode

Der Protected Mode¹ basiert auf einer mehrstufigen Speicherorganisation. Im Gegensatz zum Real Mode wird zwischen die Ebene des logischen und des physikalischen Speichers noch eine weitere Ebene gelegt. Diese Ebene - virtueller Speicher genannt - ermöglicht die Entkopplung des logischen vom physikalischen Adreßraum. Dies bedeutet, daß Anwendungen

¹Protected Mode = "geschützter Betrieb", d.h. das Betriebssystem soll auch im Falle eines Fehlverhaltens von Anwendungen stabil funktionieren

nicht mehr auf reale Speicherzellen zugreifen können, sondern nur noch auf einen abstrakten Adreßraum, der jedoch unabhängig vom tatsächlich vorhandenen Arbeitsspeicher den gesamten linearen Speicher umfaßt. Der für Anwendungen verfügbare Adreßraum wird damit weit- aus größer als im Real Mode und beträgt 4 GB. Ein weiterer Vorteil ist, daß es nun unerheb- lich ist, ob sich die genutzten Adreßbereiche auf einer Festplatte oder im Arbeitsspeicher be- finden. Ausführliche Erläuterungen zur Virtualisierung finden sich in der Literatur (z.B. [2.8]).

Der virtuelle Adreßraum ist ein wesentliches Werkzeug zum Aufbau von sogenannten Multi- taskingsystemen, da es nun möglich ist voneinander unabhängige Adreßräume zu erstellen, die separate Anwendungen enthalten können und über ein entsprechendes Prozessormanage- ment bearbeitet werden. Dabei ist anzumerken, daß Ein-Prozessorsysteme zwei Anwendungen niemals gleichzeitig bearbeiten können. Allerdings kann die Bearbeitungsgeschwindigkeit mehrerer Anwendungen so hoch sein, daß es dem Benutzer so scheint, als würden alle An- wendungen gleichzeitig bearbeitet. Genaue Erläuterungen zur Arbeitsweise im Protected Mode gibt die Literatur [2.9].

Wichtig ist jedoch, daß die virtuelle Speicherverwaltung immer einen höheren Zeitaufwand erfordert. Deshalb treten insbesondere bei Betriebssystemaufrufen, wie z.B. Interrupts, Ge- schwindigkeitseinbußen auf, da das Betriebssystem die Struktur der Schutzmechanismen be- rücksichtigen muß. Für den interruptgesteuerten Betrieb von Stromrichtern ist der Protected Mode daher weniger geeignet, da es wegen wiederholter Interrupts zu Leistungseinbußen kommt.

2.2.2 Der Interruptcontroller 8259A

Beim üblichen PC-Betrieb arbeitet der Interruptcontroller im Verborgenen. Er verwaltet exter- ne Geräte wie das Festplattenlaufwerk, die Tastatur oder die Schnittstellen. Der Anwender be- merkt davon nichts, da er nur auf die vom Betriebssystem oder der Anwendungssoftware vor- gegebenen Routinen zurückgreifen kann. Insofern ist für normalen Betrieb dieser Baustein von geringerer Bedeutung. Die Meldung von äußeren Ereignissen über Interrupts ist aber in Steuerungen und Regelungen eine zentrale Technik. Deshalb muß dieser Baustein hier ausführlicher behandelt werden.

Bild 2.4 zeigt eine schematische Darstellung eines Bausteins vom Typ 8259A [2.10]. Im AT- Modell sind zwei Bausteine des Typs 8259A in Kaskadenschaltung bereits auf der Grundplatine integriert. Dabei wirkt ein Controller direkt und der kaskadierte Controller über den Eingang IRQ2 des ersten. Damit sind genau 15 Interrupts nutzbar.

Ein Ereignis wird zunächst vom Controller registriert, der dann in Wechselwirkung mit dem Prozessor tritt. Auf diese Weise muß sich der Prozessor nicht um die Frage des Auftretens dieser Ereignisse kümmern.

Über die Leitungen IRQ 0 bis IRQ 7 können im Interrupt-Request-Register bis zu 8 Ereignisse getrennt voneinander erfaßt werden. Die nach dem AT-Modell angeschlossenen Geräte sind im Anhang A.2 aufgelistet. Das Interrupt-Mask-Register¹ filtert diejenigen Signale heraus, die an das System weitergegeben werden dürfen. Das ist insbesondere für den Betrieb in Steuerungen wichtig. Über die Änderung des Werts dieses Registers kann gezielt jede einzelne Interruptleitung (z.B. Signale des Systemtimers oder der Maus) unterdrückt werden. Im In-Service-Register sind die derzeit anhängigen Interrupts gespeichert. Die Steuerungslogik beginnt die Interruptsequenz. Für den Datenaustausch mit dem System steht ein Datenpuffer zur Verfügung.

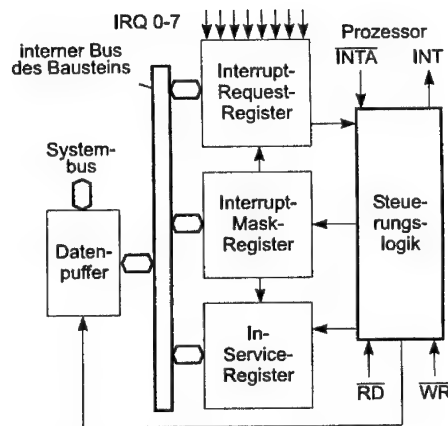


Bild 2.4: Schema des Interruptcontrollers

Die Initialisierung des Bausteins wird während des Boot-Vorgangs vom BIOS vorgenommen. Sie beinhaltet die Definition der Hardwareinterruptbereiche innerhalb der Interruptbeschreibungstabelle von DOS sowie die Festlegung von Prioritätsebenen. Nach der Bearbeitung eines Interrupts muß das "End-Of-Interrupt-Bit" des Interruptcontrollers gesetzt werden, damit die nächste Interruptanforderung eingelesen werden kann (vgl. Abschnitt 2.3.2). Eine ausführliche Beschreibung der Programmierung des Bausteins findet sich in [2.11]. Die Interruptbeschreibungstabelle von DOS enthält 256 Interruptmöglichkeiten. Alle nicht den Interruptcontrollern zugeordneten Interrupts sind Softwareinterrupts.

Während im normalen Betrieb Softwareinterrupts z.B. zum Ablesen der Systemuhr oder zum Ansteuern des Bildschirms benötigt werden, sind dies bei Steuerungen und Regelungen vor allem Hardwareinterrupts in Verbindung mit einem Timer-Baustein. Die genaue Beschreibung des Ablaufs von Hardwareinterrupts erfolgt in Abschnitt 2.3.2.

¹Engl. "to mask" = maskieren, verstecken; ein auf einer maskierten Leitung eingehendes Signal wird von System nicht beachtet und bearbeitet

2.2.3 Der Timer-Baustein 8254

Timerbausteine (Zeitwerke) dienen der Zeitmessung, der Festlegung von Zeitintervallen oder der Zählung von Ereignissen. Im AT-Modell sind zwei dieser Bausteine vorhanden. Sie übernehmen verschiedene Systemaufgaben wie z.B. die Ermittlung der Systemzeit und des Systemdatums. Beides kann mit den DOS-Befehlen "TIME" bzw. "DATE" abgefragt und modifiziert werden. Viele Anwendungen benutzen die Systemzeit, um z.B. Dateien in regelmäßigen Intervallen abzuspeichern. Auch der "Systempieps" wird mit Hilfe des Timerbausteins erzeugt. Im normalen Betrieb tritt die Wirkung dieses Bausteins jedoch nur selten zu Tage. Bei Steuerungen und Regelungen im allgemeinen und für Stromrichtersteuerungen im speziellen werden Timerbausteine für die Ausgabe von Steuersignalen zu genau definierten Zeitpunkten eingesetzt.

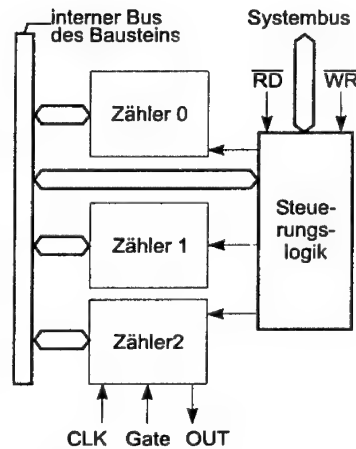


Bild 2.5: Aufbau des Zählerbausteins 8254

Bild 2.5 zeigt das Prinzipbild dieses Bausteins, der drei unabhängige 16-bit Zähler enthält, wobei nur am Zähler 2 sind auch die Steuerleitungen eingezeichnet sind. Jedes Zeitwerk kann in 6 verschiedenen Modi betrieben werden [2.12]. Für Anwendungen in der Stromrichtertechnik wird aber nur Modus 0 (Single-Shot-Betrieb) benutzt. Hier verringert der Timer den programmierten Wert bei jedem Clock-Signal "CLK" um 1. Während des Zählens liegt der Ausgang "OUT" des Zählers auf LOW-Pegel. Mit dem Erreichen des Werts "0", wechselt der Ausgang auf HIGH. Der HIGH-Zustand kann in Steuerungen zum Auslösen von Interrupts benutzt werden. Mit dem Gate-Signal kann das Zeitwerk angehalten werden. Das Setzen der Zählintervalle am Timer erfolgt durch das Schreiben zweier aufeinanderfolgender 8-Bit-Werte über den Systembus.

Die Initialisierung der beiden zum AT-Modell gehörigen Bausteine nimmt das BIOS vor. Mittlerweile sind diese Bausteine im Chip-Set integriert und für ihre Aufgaben fest verdrahtet. Sie bieten daher wenig Eingriffsmöglichkeiten und Flexibilität [2.13]. Deshalb werden zusätzliche Einsteckkarten mit Zählern benutzt, um die notwendigen Zeitintervalle für die Steuerung und Regelung zu erzeugen.

2.2.4 Der DMA-Controller

Der DMA (Direct-Memory-Access) -Controller trägt erheblich zur Effizienz von PC-Systemen bei. Normalerweise ist die CPU an der Datenübertragung von und zu externen Geräten beteiligt. Für die Übertragszeit ist das System sozusagen gelähmt, was insbesondere bei großen Datenmengen erhebliche Geschwindigkeitseinbußen mit sich bringt.

Genau an diesem Punkt setzt der DMA-Controller ein. Er schaltet die sogenannten DMA-Kanäle frei, und verbindet damit die Datenquelle direkt mit dem Arbeitsspeicher. Die Freischaltung erfolgt aufgrund einer "DMA-Anforderung", die ein externes Gerät oder Bauteil auslöst. Der Datenbus ist dann nicht vom Datentransfer betroffen, so daß das übrige System weiter die aktuelle Routine bearbeiten kann. Es ist somit eine Parallelisierung bezüglich des Datentransfers eingetreten. Bild 2.6 verdeutlicht dies. Im Prinzip ist jeder der vier im AT-Modell vorhandenen DMA-Kanäle verfügbar, meist sind jedoch bestimmte Kanäle bereits über die Systeminitialisierung vergeben. So dient der DMA-Kanal 0 beispielsweise der dynamischen Speicherauffrischung.

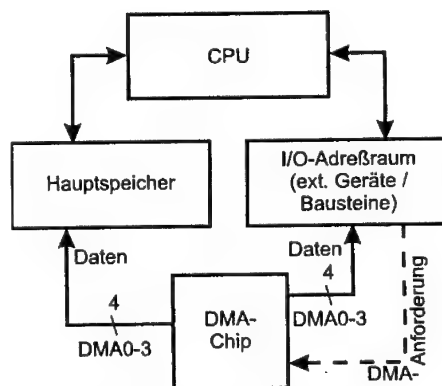


Bild 2.6: DMA-Controller im System

Die Programmierung des DMA-Controllers ist vergleichsweise umfangreich und beeinflusst zahlreiche Systemfunktionen. Es ist daher für unerfahrene Programmierer nicht ratsam, diesen Baustein selbst zu programmieren. Verschiedene Entwicklungsumgebungen von Programmiersprachen (z.B. LabWindows/CVI) enthalten jedoch bereits Hilfsmittel, mit denen der Zugriff auf DMA-Kanäle vereinfacht wird. Eine genaue Beschreibung der Programmierung findet sich in [2.14]. Im beschriebenen System wird ein DMA-Kanal für die Übertragung der Meßdaten benutzt. Die Messungen müssen zu bestimmten Zeitpunkten stattfinden. Aber gerade zu diesen Zeitpunkten ist der Rechner voll beschäftigt.

2.2.5 Einsteckplätze (Slots)

Die Grundplatine (Motherboard) des AT-Systems stellt Steckplätze zur Erweiterung der Systemfunktion zur Verfügung. Diese sind spezielle Anschlüsse an den Adreß- und Datenbus sowie an die Steuerleitungen (Schreib-Lese-Anforderung, Interrupt- oder DMA-Anforderung, u.ä.). Die Anschlüsse sind auf Bussysteme normiert, die auch in anderen PC-Systemen (z.B. Macintosh) zum Einsatz kommen. Im AT-Modell sind der ältere ISA (Industrial Standard Architecture) -Bus und der modernere PCI (Peripheral Component Interface) -Bus üblich (s. Bild 2.7). Die Einsteckkarten müssen dafür entsprechende Umsetzer enthalten. Die Zugriffsmöglichkeiten auf solche Baugruppen werden in Abschnitt 2.3 beschrieben.

Der ISA-Bus ist bei Systemen, die nur über diesen Bus verfügen, über die ISA-Bus-Brücke direkt an den Systembus angeschlossen. Bei den heute üblichen Motherboards mit PCI-Bus ist der ISA-Bus mit einer geeigneten Schnittstelle auf den PCI-Bus aufgesetzt. Während der ISA-Bus nur mit 8,33 MHz getaktet ist, arbeitet der PCI-Bus mit 66 oder 100 MHz [2.15]. Ob-

wohl für den ISA-Bus eine große Zahl an billigen Einsteckkarten für fast jeden erdenklichen Anwendungsfall existiert, wird die Zukunft dem schnelleren PCI-Bus gehören, da er neben der höheren Taktrate auch über einen 32-Bit breiten Daten- und Adreßbus verfügt und damit die ganze Leistungsfähigkeit heutiger Prozessoren ausnutzen kann. Die Leistungserhöhung geht aber mit einer weitaus komplexeren Programmierung und Hardwareanbindung der Einsteckkarten einher. Trotzdem ist in der neuesten Spezifikation von Personal Computern der ISA-Bus kein zwingender Bestandteil mehr [2.16].

Für Steuerungs- und Regelungsaufgaben ist eine Mehrzweck-Einsteckkarte empfehlenswert. Sie kann analoge Signale in digitale Daten wandeln, digitale Signale ausgeben und einlesen sowie Zeitintervalle angeben. Im weiter unten beschriebenen Steuer-PC wurde eine Karte mit PCI-Bus verwendet. Trotz ihres größeren Verwaltungsaufwandes bietet sie eine höhere Verarbeitungsgeschwindigkeit als eine entsprechende ISA-Karte.

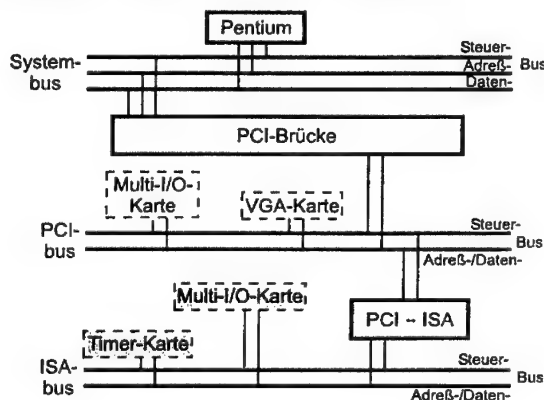


Bild 2.7: Aufbau der Erweiterungssteckplätze mit ISA- und PCI-Bus eines Pentiumsystems

2.2.6 Der digitale Ein-/Ausgabebaustein 8255

Mit diesem Baustein können digitale Signale ausgegeben oder eingelesen werden. Er ist zwar nicht mehr Bestandteil des normalen AT-Modells, wird an dieser Stelle aber dennoch beschrieben, da er als Komponente auf Multi-I/O-Karten häufig anzutreffen ist.

Bild 2.8 zeigt den Aufbau des Bausteins. Er umfaßt drei unabhängige 8-Bit Ports, wovon Port C wiederum in zwei 4-Bit Ports unterteilt ist. In der Betriebsart 0¹ sind alle drei Ports unabhängig voneinander

als Eingänge oder Ausgänge konfigurierbar. Zur Ausgabe der Ansteuersignale für Stromrichter mit B6-Brücken sind nach Abschnitt 5.1.2 mindestens 6 Signale erforderlich. Außerdem ist es in Steuerungen und Regelungen häufig notwendig noch weitere Signale aus der Prüfstandsumgebung, wie z.B. den Zustand von Schützen usw., auszuwerten, so daß nur die Betriebsart 0 für eine ausreichende Anzahl an Ein- oder Ausgabeleitungen sorgt.

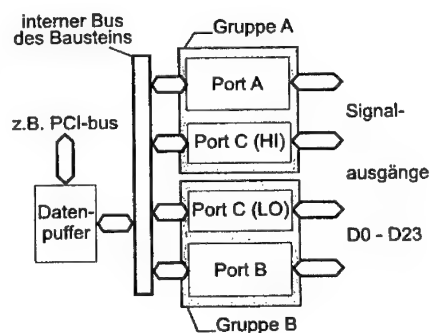


Bild 2.8: Prinzipbild des Bausteins 8255

2.2.7 Tastaturcontroller (8042)

Die Tastatur ist im Normalbetrieb eines der wichtigsten Eingabegeräte, da jedes Softwareprogramm meist über eine Tastatur programmiert wird, bevor es dem Benutzer als Anwendung zur Verfügung steht. Aber nicht nur die Herstellung von Software, sondern auch deren Bedienung erfordert im allgemeinen eine Tastatur, mit der Texte eingegeben, Tabellen erstellt oder Datenbanken aufgefüllt werden können. Für Steuerungen und Regelungen ist die Tastatur oft in einer auf die notwendigen Bedienknöpfe reduzierten Form vorhanden. In AT-Systemen ist eine sogenannte Multifunktions- (MF II-) Tastatur bereits Bestandteil und kann für Eingaben benutzt werden.

¹ Die Betriebsarten 1 und 2 dienen der parallelen Datenübertragung. Beide sind in der Literatur [2.17] beschrieben. Dazu werden die drei Ports in 2 Gruppen aufgeteilt. Im älteren XT-Modell wird dieser Baustein zum Aufbau der parallelen Schnittstelle herangezogen.

Der exakte Aufbau und die Arbeitsweise des Controllers kann in der Literatur (z.B. [2.18]) nachgeschlagen werden. An dieser Stelle werden nur die für die Steuerung und Regelung von Stromrichtern notwendigen Aspekte beschrieben.

Unabhängig von der Tastenbelegung einer Tastatur wird vom Tastaturcontroller der sogenannte "Scancode" ermittelt, der sich auf die jeweils gedrückte ("Make-Code") oder losgelassene ("Break-Code") Taste bezieht und Werte von 0 bis 255 annehmen kann. Allgemein gilt: Break-Code = Make-Code+128. Eine Liste des Scancodes und der zugeordneten Tasten einer deutschen Tastatur ist im Anhang A.4 angegeben. Aufgrund des Scancodes kann mit Hilfe der länderspezifischen Tastaturtreibersoftware die gedrückte Taste ermittelt und ggf. weiterverarbeitet werden.

Im allgemeinen werden Interaktionen zwischen System und Tastatur von Hard- und Software-interrupts organisiert. Da beide Arten von Programmunterbrechungen (siehe Abschnitt 2.3.2) ggf. den Stromrichterbetrieb gefährden können, muß die Tastatur ohne Interrupts mit direktem Lesen aus den Portadressen des Controllers gesteuert werden. Bild 2.9 zeigt die wichtigsten Bestandteile dieses Bausteins. Von den jeweils zwei Puffern und Registern zu je 8 Bit sind im weiteren nur je eines von Interesse: der Ausgabepuffer und das Statusregister. Mit Bit 0 des Statusregisters wird geprüft, ob im Ausgabepuffer ein vollständiges Byte zur Verfügung steht, das ausgelesen werden kann. Es stellt den Scancode einer betätigten Taste dar, der dann in der Anwendung entsprechend weiterverarbeitet wird.

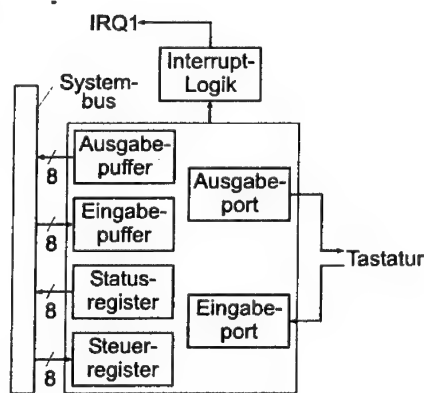


Bild 2.9: Schema des Tastaturcontrollers

2.2.8 Speicher

Dieser Teil eines PC-Systems kann maßgeblich die Leistung beeinflussen, da er bei fast allen Befehlen benötigt wird. Zur Begriffserklärung sei an dieser Stelle auf die Literatur [2.19] verwiesen. In neueren PC-Systemen stehen zwei Arten von Speichern mit unterschiedlichen Eigenschaften und Einsatzgebieten zur Verfügung.

2.2.8.1 Cache

Der Cachespeicher stellt in gewisser Hinsicht den Vorratsschrank des Prozessors dar. In diesem Speicher werden diejenigen Daten und Befehle zwischengespeichert, die mehrmals benötigt werden. Der Cache selbst wird in internen (auch L1-) und externen (L2-)Cache unterteilt. Der interne Cache ist bereits in den Prozessor integriert (siehe Bild 2.1), während der externe Cache mittels diskreter Speicherchips oder montierter Module in das System eingefügt wird. Da die Funktionsweise jedoch dieselbe ist, wird im allgemeinen nicht zwischen intern und extern unterschieden. Die Größe des L1-Caches ist abhängig vom Prozessortyp. Sie reicht von 16 kB bei Pentium60-Prozessoren bis hin zu 64 kB bei neuesten Prozessoren. Die Verwaltung des Caches obliegt ausschließlich dem Prozessor, so daß das Betriebssystem keine Einflußmöglichkeit darauf hat. Die Organisation, Struktur und Betriebsarten des Caches sind in [2.20] ausführlich beschrieben.

Der Unterschied zu herkömmlichem Speicher besteht darin, daß durch die Cachestruktur sehr schnelle Speicherzugriffe möglich sind. Die Zugriffszeiten bewegen sich dabei um den Faktor 1/3 bis 1/5 unterhalb derer normaler Speicherbausteine. Diese Effizienzsteigerung erhöht sich noch dadurch, daß nur häufig benötigte Daten und Befehle in den Cache gelangen. Ein schnelles PC-System sollte deshalb unabhängig von dessen Verwendung mit ausreichendem Cache ausgestattet sein.

2.2.8.2 Arbeitsspeicher (RAM)

Im Arbeitsspeicher sind die Anwendungen oder Teile davon abgelegt. Jeder Befehl oder jedes Datum muß also immer vom Arbeitsspeicher zum Prozessor gelangen. So stellen Speichereinheiten mit zu großen Zugriffszeiten einen unüberwindlichen Flaschenhals dar. Obwohl der Prozessor ggf. weitaus mehr Daten verarbeiten könnte, ist die Gesamtausnutzung gering, da der Speicher die erforderlichen Daten nicht in ausreichender Geschwindigkeit zur Verfügung stellt.

Bei umfangreichen Anwendungen kann es auch vorkommen, daß Teile des Programmcodes nicht mehr im Arbeitsspeicher abgelegt werden können, sondern auf der Festplatte zwischengespeichert werden. Da das Verhältnis der Zugriffszeiten zwischen Speicher und Festplatten bei etwa 1:1000 liegt, bedeutet eine solche Vorgehensweise eine weitere Minderung der Rechenleistung. Selbst für den normalen Betrieb sollte ein PC demnach mit ausreichend Arbeitsspeicher ausgestattet sein, wobei die Zugriffszeit auf den Speicher selbst gering sein sollten (< 60 ns). Aktuelle PC sind mittlerweile mit Speichermodulen ausgestattet, die Zugriffszeiten bis zu 10 ns zulassen.

2.3 Zusammenwirken der Bausteine im AT-System

2.3.1 Hardware-Programmierung

Für Steuerungen und Regelungen spielt die Programmierung der internen Bausteine eine große Rolle. Das AT-Modell sieht deshalb für jeden Baustein einen reservierten Bereich des I/O-Adreßraums vor, welcher der Kommunikation mit dem Baustein und dessen Programmierung dient. Die jeweils niedrigstwertige Adresse solcher Adreßbereiche wird Basisadresse oder Portadresse genannt. Die Adreßbereiche umfassen meist 8 oder 16 Byte, die aber nicht voll genutzt werden müssen. Baugruppen, die nicht im AT-Modell vorgesehen sind (z.B. Einsteckkarten), können darüber hinaus zusätzliche Speicherbereiche belegen. Innerhalb ihres Adreßbereichs können sie wiederum Teile davon anderen programmierbaren Bausteinen zuteilen. Auf diese Weise können mehrere solcher Bauteile auf einer Einsteckkarte eingesetzt werden. Tabelle 2.1 zeigt die Adreßbereiche ausgewählter Bausteine des AT-Modells.

Den Registern der Peripherie, wie z.B. dem Interrupt-Mask-Register des Interrupt-Controllers, sind relative Adressen innerhalb ihres Bereichs zugeordnet. Ist also die Basisadresse erst einmal definiert, so sind es auch die Adressen der Register. Die Abweichung der Registeradressen von der Basisadresse wird Offset genannt. Die Adressierung ist also ähnlich wie im Real Mode. Der einzige Unterschied ist, daß die Adreßbereiche der Bausteine nicht 64 kB umfassen, sondern wesentlich kleiner sind. Der Schreibvorgang wird z.B. in der Programmiersprache C mit dem Befehl "outp (Adresse, Wert)" und der Lesevorgang mit dem Befehl "inp (Adresse)" realisiert. In anderen Programmiersprachen existieren entsprechende Befehle.

Tabelle 2.1: Bausteine und deren Basisadressen im AT-Modell (Auszug)

Baustein	Adressbereich (hex.)
Interruptcontroller 1	0x20 - 0x3F
Zeitgeber (System)	0x40 - 0x5F
serielle Schnittstelle 1	0x3F8 - 0x3FF
Tastatur	0x60 - 0x6F
Netzwerkkarte	0x360 - 0x36F

2.3.2 Interrupts

Interrupts sind ein verbreitetes Mittel, um Bausteine, Geräte oder Baugruppen, die nicht fortwährend genutzt werden, effizient in das System einzubinden. Im weiteren werden Hardware-Interrupts nur noch Interrupts genannt. Systeme, die keine Interruptmöglichkeiten

besitzen, müssen ihre Umgebung in festgelegten Zeitspannen komplett selbständig erfassen, um Zustandsänderungen zu erkennen (Polling-Systeme). Dadurch kann eine erhebliche Bindung von Rechenleistung entstehen.

Interrupts zwingen das System zu Reaktionen im Programmablauf. Dabei greifen die oben erwähnten Bausteine in komplexer Weise aufeinander zu. Allerdings treten beim PC insbesondere wegen der vielfältigen Interruptmöglichkeiten Probleme auf, die bei vergleichbaren Mikrocontrollern keine Rolle spielen. Diese Probleme lassen sich durch Eingriffe in die Software und auch die Hardware teilweise kompensieren. Im folgenden soll hier nur auf den Ablauf eines Interrupts eingegangen werden (siehe Bild 2.10). Die spezielle Realisierung der Interruptroutine wird später anhand der konkreten Anwendung genauer beleuchtet.

Wird der CPU ein Interrupt durch ein HI-Signal am Anschlußpunkt INTR gemeldet, beendet der Prozessor zunächst den aktuellen Befehl mit der Nummer n. Dann gibt er das Interrupt-Acknowledge-Signal INTA an den Interrupt-Controller zurück, der ihm daraufhin über den Bus die Interruptnummer mitteilt. Anschließend speichert der Prozessor seine Register auf dem Stack ab und löscht im Flagregister (EFLAGS, s. Anhang A.1) des Prozessors das Interrupt-flag IF. Erst dann zeigt das Befehlsregister EIP der CPU auf den Anfang der

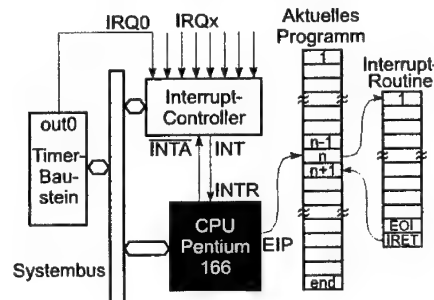


Bild 2.10: Vorgang eines Interrupts am Beispiel eines Interrupts durch den Systemtimer (EIP: Befehlsregister)

durch die Interruptnummer festgelegten Routine. Der als Abschluß stets notwendige Interrupt-Return-Befehl IRET sorgt für das Laden der alten Register vom Stack. Damit zeigt das EIP-Register wieder auf das unterbrochene Programm, das mit dem nächsten Befehl (n+1) fortgesetzt wird. Die Interruptroutine wird also in das Programm zwischen die Schritte n und n+1 eingefügt. Wie in Abschnitt 2.2.2 beschrieben, ist es vor dem Verlassen der Interrupt-Service-Routine zwingend notwendig, das "End-Of-Interrupt-Bit" des Interrupt-Controllers zu setzen.

Treten während der Bearbeitung Interrupts gleicher oder niedrigerer Priorität auf, so werden diese erst nach Abschluß des Interrupts in der Reihenfolge ihrer Priorität bearbeitet. Tritt ein Interrupt höherer Priorität auf, so wird der laufende Interrupt zugunsten des neuen unterbrochen.

Anhand der vom Interrupt-Controller übermittelten Interruptnummer wertet die CPU die sogenannte Interrupt-Descriptor-Tabelle (IDT) aus, welche die 256 Anfangsadressen der Interruptroutinen enthält. Die IDT nimmt im AT-Modell immer den Adreßbereich $0_h - 3FF_h = 1\text{kB}$ im Segment 0 ein und ist einer der beiden fest der CPU zugeordneten Speicherbereiche des AT-Modells [2.21]¹.

Der beschriebene Ablauf eines Interrupts benötigt vor allem wegen der Sicherung und des Zurückholens von Registerinhalten der CPU eine bestimmte Mindestzeit. Sie tritt auch dann auf, wenn von der Interruptroutine keine Befehle ausgeführt werden. Da die Anzahl der Maschinenbefehle, die für eine Interruptbearbeitung notwendig sind, konstant ist, hängt die Mindestzeit vor allem von der Rechengeschwindigkeit der CPU ab. Tabelle 2.2 zeigt diese Zeitdauern für 3 verschiedene Konfigurationen.

Tabelle 2.2: Vergleich der Mindestdauer einer Interruptroutine

Konfiguration	Dauer [μs]
i486, 66 MHz	16
P166, 166 MHz	6
PII, 400 MHz	3

Bei der Steuerung von Stromrichtern können Vorgänge auftreten, die eine Folge von Interruptroutinen verlangen, deren Abstände kürzer als die Mindestzeiten sind. Für solche Fälle werden Sonderrouinen benutzt, deren Aufbau und Funktion in dieser Arbeit behandelt werden. Die allgemeine Struktur von Interruptroutinen wird in Anhang B beschrieben.

2.4 Betriebssysteme

Das Betriebssystem sorgt für eine Oberfläche zur Verwaltung und Programmierung des Systems. Das beinhaltet nicht nur eigene Funktionen zur Programm- und Hardwareverwaltung, sondern auch die Möglichkeit, über BIOS-Funktionen auf die Hardware zugreifen zu können. Bild 2.11 zeigt schematisch die Zugriffsmöglichkeiten von BIOS, Betriebssystem, Hardware und Anwendungen.

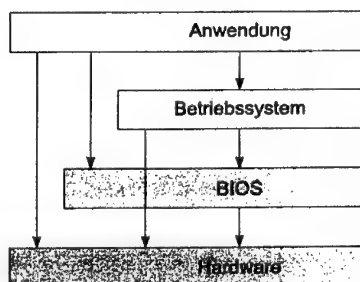


Bild 2.11: Zugriffsmöglichkeiten von Betriebssystem, Anwendungen, Hardware und BIOS im AT-Modell

¹Der zweite feste Adreßbereich im AT-Modell ist der Speicherbereich für die Einsprungadresse der Startroutine des BIOS (Adresse $FFF0_h$ im Segment $F000_h$)

Im normalen Betrieb reichen die Betriebssystem- und BIOS-Funktionen für die meisten Anwendungen aus. Bei interruptgesteuertem Stromrichterbetrieb sind jedoch Eingriffe in das Betriebssystem (z.B. die Programmierung eigener Interruptroutinen) notwendig. Dies ist allerdings nicht in jedem Betriebssystem ohne weiteres möglich.

2.4.1 Singletasking Systeme

Das Betriebssystem DOS ist das bekannteste Beispiel für solche Systeme, die nur jeweils eine Anwendung bearbeiten können. Es kann im Real Mode lediglich 640 kB für Anwendungen verwalten. Der Bereich von 640 kB bis 1 MB ("hoher Speicherbereich") wird für Systemdaten, wie z.B. das BIOS-ROM oder die Bildschirmdatei, genutzt. Mit Hilfe von Zusatzprogrammen ist es jedoch möglich einen Teil der Betriebssystemdateien dort abzuspeichern, um mehr Speicherplatz für Anwendungen zu erhalten. Im Protected Mode dagegen könnte DOS mit Hilfe weiterer Dienstprogramme die gesamte Breite des Adreßbusses zur Speicherverwaltung nutzen. Allerdings hat hier der höhere Aufwand auch eine niedrigere nutzbare Rechenleistung zur Folge, was sich besonders bei Systemaufrufen, wie z.B. Interrupts, auswirkt. Sowohl im Real Mode wie auch im Protected Mode werden externe Geräte direkt über deren Basisadressen programmiert werden.

Für die Anwendung als Stromrichtersteuerung ist ein Adreßraum von 1 MB völlig ausreichend, so daß DOS im Real Mode dafür geeignet ist. Der für Offline-Simulationen erhöhte Bedarf an Speicherplatz kann dann durch Umschalten in den Protected Mode bereit gestellt werden.

2.4.2 Multitasking Systeme

Diese Gruppe von Betriebssystemen (z.B. WindowsNT, Windows98 oder Linux) kann verschiedene Anwendungen oder Aufgaben (=Tasks) "gleichzeitig" bearbeiten. Echte Gleichzeitigkeit kann jedoch nur auf Systemen, die mehrere Prozessoren enthalten, erreicht werden. Systeme mit nur einem Prozessor hingegen versuchen durch spezielle Zuteilungsregeln die Rechenleistung der CPU so zu verteilen, daß der Anwender den Eindruck gewinnt, alle Anwendungen würden gleichzeitig bearbeitet.

Multitasking-Betriebssysteme betreiben den Prozessor ausschließlich im Protected Mode, da nur diese Betriebsart die nötigen Voraussetzungen zum Schutz des Betriebssystems einerseits und der Anwendungen untereinander andererseits bietet. Für die vorliegende Aufgabe sind solche Betriebssysteme aber ungeeignet, da die Bearbeitungszeit von Interrupts eines laufenden Tasks nur in einem Zeitintervall angegeben werden kann (z.B. wegen Prioritätenverwaltung,

Interruptverwaltung, Zeit zur Umschaltung zwischen den Tasks etc.) [2.22]. Bei der Steuerung von Stromrichtern sind aber genau festgelegte Zeitpunkte für die Umschaltung der Leistungshalbleiter entscheidend.

2.4.3 Echtzeitsysteme

Der Vollständigkeit halber seien noch diejenigen Systeme aufgeführt, die mit speziellen Hilfsprogrammen einen Betriebssystemkernel so abändern, daß er Echtzeitkriterien standhält oder sogar den Kernel selbst ersetzen. An je einem Beispiel für jede Gruppe wird deren Eignung für den Betrieb von Stromrichtern kurz erläutert.

Zur ersten Gruppe zählt beispielsweise das "Industrial Toolkit" der Firma Kithara für Windows NT [2.22]. Es unterstützt den direkten Zugriff auf die Hardware-Geräte eines PC (z.B. Timer, Interrupt-Controller, etc.), der bei diesem Betriebssystem sonst nicht erlaubt ist. Unter diesem Gesichtspunkt ist es durchaus zur Regelung von Stromrichtern geeignet. Da dieses System kein Betriebssystem als solches darstellt sondern nur auf Windows NT aufsetzt, gelten nach wie vor die Unsicherheiten in Bezug auf Taskwechsel bzw. Interruptbearbeitung wie sie in Abschnitt 2.4.2 beschrieben sind, sodaß es für die gestellte Aufgabe nicht in Frage kommt.

Zur zweiten Gruppe gehört z.B. das System QNX [2.23]. Es wird beim Boot-Vorgang anstelle eines herkömmlichen Betriebssystems geladen. Es ist darauf ausgerichtet, Prozesse in festgelegten Zeiträumen zu bearbeiten. Die einzelnen Prozesse lassen sich priorisieren und haben mehrere Möglichkeiten, Daten mit anderen Prozessen auszutauschen. Dazu sind der Scheduling-Dienst für die Ablaufplanung der Prozesse und Nachrichten-Dienst zur Verarbeitung der zwischen ihnen gesendeten Daten erforderlich. Der Nachteil solcher Systeme ist jedoch, daß sich auch hier durch die internen Vorgänge der Interrupterkennung eine zeitliche Verzögerung ergibt. Sie liegen im Bereich weniger Mikrosekunden (z.B. 3.3 μ s für Pentium166 [2.23]). Es existiert aber noch eine zusätzliche Verzögerung, die durch den Wechsel von einem Prozeß zum anderen entsteht. Sie liegt ebenfalls im Bereich einiger Mikrosekunden (z.B. 4.7 μ s für Pentium166). Allerdings bieten nur sehr wenige Hersteller von PC-Einsteckkarten entsprechende Treiber für QNX an, sodaß die Auswahl der Einsteckkarten stark eingeschränkt ist bzw. die Treiber meist zusätzlich selbst programmiert werden müssen. Ohne geeignete Treiber können aber beispielsweise die Zeitwerke von Multi-I/O-Karten nicht genutzt werden. Dadurch stehen dann lediglich die Timer-Funktionen des QNX-Systems zur Verfügung, deren Auflösung zwischen 500 μ s und 50 ms eingestellt werden kann. Für die Realisierung von Pulsperiodendauern von 100 μ s kann QNX damit nur eingeschränkt verwendet werden.

3 Antriebsprüfstand

Bild 3.1 zeigt den prinzipiellen Aufbau des Prüfstandes, mit dem die in Abschnitt 1.3 beschriebenen Untersuchungen durchgeführt werden können. Eine dreiphasige Asynchronmaschine wird von einem Wechselrichter gespeist, der an einem Gleichspannungszwischenkreis angeschlossen ist. Eine ungesteuerte B6-Brücke stellt die Verbindung zum Drehstromnetz her. Die Ansteuersignale für den Wechselrichter werden in einem PC (AT-Modell) berechnet und ausgegeben. Über Einsteckkarten kann der PC verschiedene Betriebsparameter, wie z.B. die Zwischenkreisgrößen oder die Motorströme, aufnehmen. Die Last besteht aus einem geregelten, rückspisefähigen Drehstrom-Servoantrieb.

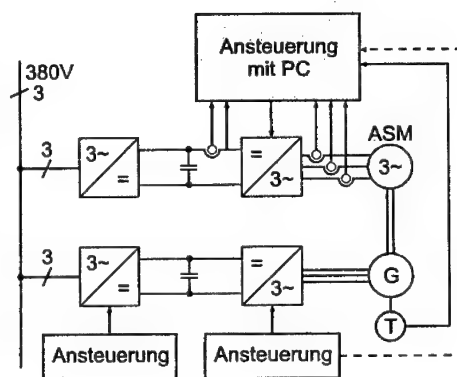


Bild 3.1:
Übersichtsschaltplan des Prüfstandes

3.1 Antriebsmaschine

Es wurde eine handelsübliche Asynchronmaschine für Frequenzumrichterbetrieb ausgewählt. Tabelle 3.1 zeigt die Daten des Leistungsschildes. Die durch den Schaltbetrieb hervorgerufene höhere Belastung der Windungsisolation ist bei dieser Maschine durch eine besondere Isolation berücksichtigt.

Tabelle 3.1: Leistungsdaten der verwendeten Maschine (Asynchronmotor)

Größe	Wert
DIN-Baugruppe	160 L
Nennspannung U_N	380 V (Y)
Nennstrom I_N	38,5 A
Stellbereich f_N	17 - 50 Hz
Drehzahl im Bemessungspunkt n_N	1425 min ⁻¹
Nennleistung P_N	18,5 kW

3.2 Last

Als Belastungsmaschine wurde ein Servoantrieb der Fa. Bosch (Typ "Servodyn") verwendet. Die technischen Daten des Bremsantriebs zeigt Tabelle 3.2. Er besteht aus einem geregelten Gleichrichter (VM 60/100 R-T), einem Spannungszwischenkreis, dem Steuergerät (SM 50/100-T) sowie der Maschine (SD-B6.960.015) selbst. Der Antrieb wird mit konstantem Bremsmoment betrieben, wobei das minimal einstellbare Bremsmoment auf 7% des maximalen Drehmoments begrenzt ist. Um einen möglichen Motorbetrieb des Bremsantriebs zu verhindern, wurde bei den Versuchen die Solldrehzahl fest auf $n = 0 \text{ min}^{-1}$ eingestellt.

Tabelle 3.2: Technische Daten des Lastantriebs (Bosch)

Größe	Wert
Netz-Anschlußspannung	380 - 415 V (3~)
Netz-Nennstrom	72 A
Maschinennennstrom	50 A
Maschinenspitzenstrom	100 A
Zwischenkreisspannung	460 - 700 V
Nenn Drehzahl	1500 min^{-1}
Stillstands Drehmoment	96 Nm

3.3 Antriebsstromrichter

Bild 3.2 zeigt einen Plan des Antriebsstromrichters. Er wurde mit Modulen der Fa. Semikron entsprechend den Daten des vorliegenden Asynchronmotors (siehe Abschnitt 3.1)

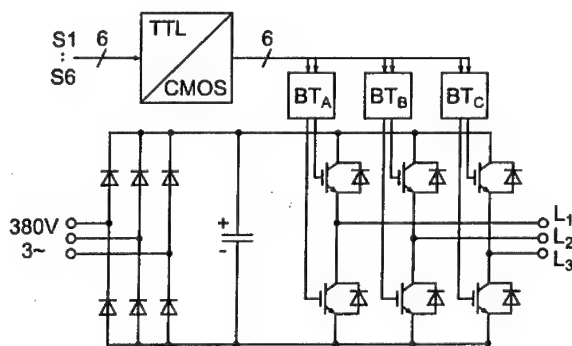


Bild 3.2: Schaltplan der Leistungsstufe mit Gleichrichter, Spannungszwischenkreis sowie des Wechselrichters mit dessen Ansteuermodulen (BT_{A,C})

dimensioniert und aufgebaut. Der am Drehstromnetz angeschlossene Gleichrichter besteht aus einer ungesteuerten B6-Brücke (SKD 110/12) und speist einen Spannungszwischenkreis, der mit einer Kapazität von 4400 µF gestützt wird. Die IGBT-Halbbrückenmodule des Wechselrichters (SKM 145 GB 123 D) sind bereits auf

einem Kühlkörper montiert, auf dem sich auch der Gleichrichter befindet. Die IGBT werden von Halbbrücken-Treibermodulen (SKHI 22) angesteuert. Diese Module erzwingen auch die für den Pulsbetrieb von Halbbrücken notwendige Sicherheitszeit, in der während des Umschaltvorgangs beide Schalter einer Halbbrücke ausgeschaltet sind. Für die nähere Beschreibung der Halbleiter sowie der Treibermodule wird auf die entsprechenden Datenblätter verwiesen ([3.1], [3.2], [3.3]). Im Bild ist noch ein Pegelumsetzer von TTL nach CMOS eingezeichnet, der allerdings notwendig ist, weil die verwendete PC-Einsteckkarte lediglich Signale auf TTL-Pegel ausgeben kann.

Bild 3.3 zeigt die Laborinstallation des Gleich- und Wechselrichters mit den notwendigen Zusatzbauteilen, wie z.B. Sicherungen, Schütze, etc., mit dem nachfolgend beschriebenen PC als Steuergerät.

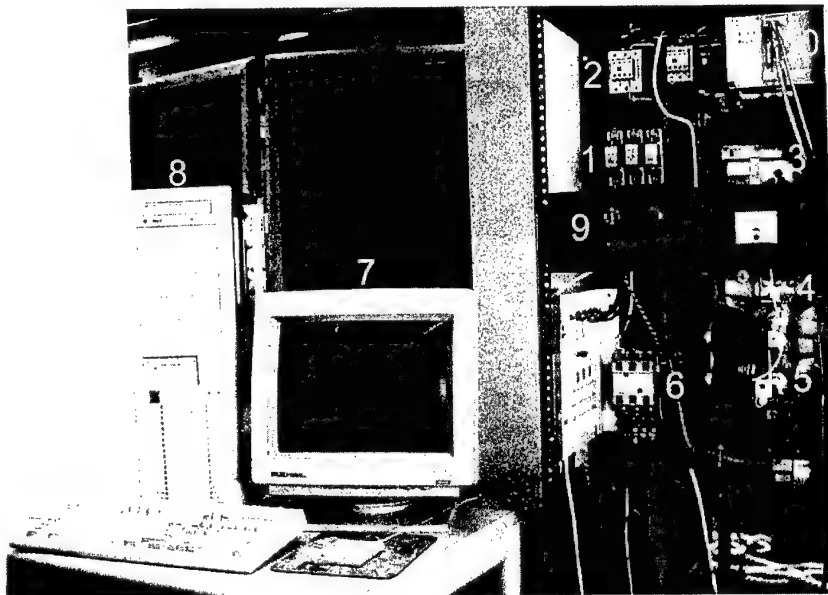


Bild 3.3: Laboraufbau (1 = Sicherungen; 2 = Netzschütze; 3 = Kommutierungs-drossel für Gleichrichter; 4 = Gleichrichter; 5 = Wechselrichter; 6 = Motorschütz; 7 = Monitor; 8 = PC; 9 = EIN-/AUS-Taster mit Leuchtanzeige; 10 = Signalkonditionierung)

3.4 PC für Steuerung und Regelung

Der PC (siehe Bild 3.4) enthält neben seiner üblichen Grundausstattung (Laufwerke, Motherboard, etc.) auch zusätzliche Mehrzweck-Einsteckkarten, welche die fehlenden Funktionen bereitstellen. Im folgenden wird nun beides beschrieben. Als zusätzliche Ein- und Ausgabegeräte dienen die Tastatur und der Monitor.

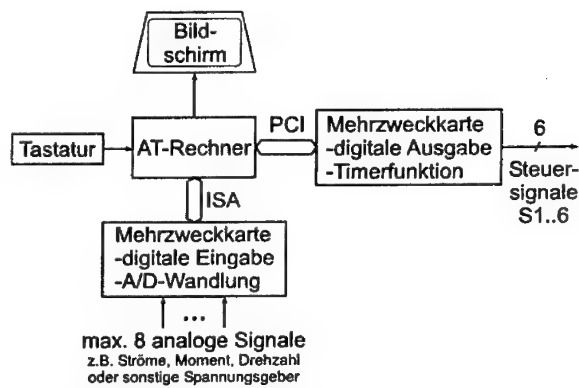


Bild 3.4: Prinzipbild eines PC, der für die Steuerung von Stromrichtern geeignet ist

3.4.1 Personal Computer (AT)

Die Literatur ([3.5]) zeigt, daß Prozessoren für AT-Rechner zwar die gleiche Funktion besitzen, unterschiedliche Implementierungen dieser Funktion jedoch zu verschiedenen Rechenleistungen führen. So ist z.B. die Integer-Recheneinheit bei Prozessoren der Fa. AMD schneller als das Pendant von Intel, wobei Intels Fließkomma-Arithmetiksektion die aller anderen Hersteller in den Schatten stellt.

Tabelle 3.3: Zeitbedarf für Berechnungsroutinen von Pulsverfahren mit verschiedenen Systemen im Real Mode

Bei stromrichtergesteuerten Maschinen sind aufgrund der wiederholten Auswertung von trigonometrischen Funktionen viele Fließkommaoperationen zu erwarten. Durch die Vorgabe flexibler Antriebsparameter (z.B. Ausgangsfrequenz) müssen wichtige Größen als Fließkommazahl definiert werden, was die Bedeutung einer schnellen Fließkomma-Einheit unter-

Pulsverfahren (System)	Dauer
Sinusmodulation der Einzelphasen (Pentium 166 MHz)	5,22 μ s
dto. (Pentium 233 MHz)	3,68 μ s
dto. (Pentium II - 400 MHz)	1,43 μ s
Zweiphasiges Pulsen 1. Art (Pentium 166 MHz)	6,1 μ s
dto. (Pentium 233 MHz)	4,34 μ s
dto. (Pentium II - 400 MHz)	1,71 μ s

streicht. Tabelle 3.3 enthält Messungen zum Zeitverbrauch von Berechnungsroutinen wie sie für zwei ausgewählte Pulsverfahren, nämlich das sogenannte "Asymmetric Regular Sampling" und "Zweiphasiges Pulsen 1. Art" (siehe Abschnitt 4.3.6) benötigt werden. Auch ältere Systeme, wie z.B. ein Rechner mit Pentium166-CPU, benötigen lediglich ca. 5 µs für die Berechnung eines Pulsabschnitts.

Wegen der komplexen Strukturen eines AT-Rechners können auch z.B. der Chip-Satz (vgl. Abschnitt 2) oder der Speicher die Systemleistung des PC entscheidend mindern. Die Fachliteratur (z.B. [3.7]) hält zum Thema Systemleistung immer wieder aktuelle Ergebnisse bereit. Dies kann als Anhaltspunkt für die Beschaffung eines geeigneten Gerätes dienen.

Für diese Arbeit wurde ein PC nach den obigen Gesichtspunkten konfiguriert. Er besteht aus folgenden, anwendungsrelevanten Komponenten:

Prozessor:	Pentium II
Taktrate:	400 MHz
Cache:	jeweils 16 kB für Befehle und für Daten (Taktung: 400 MHz)
Arbeitsspeicher:	128 MB SDRAM
Zugriffszeit:	10 ns
Motherboard:	Gigabyte GA-6BXS ATX

3.4.2 Zusatzkarten

Für die zwei wichtigsten Funktionen der vorliegenden Steuerungs- und Regelungsaufgabe, nämlich die Analog-digital-Wandlung und Zeitfunktionen ("Wecker"), werden zusätzliche Einsteckkarten benötigt. Deren Auswahlkriterien werden nachfolgend beschrieben.

3.4.2.1 Genauigkeit von Zeitfunktionen

Für diese Arbeit ist eine Pulsfrequenz von mindestens 5 kHz vorgesehen. Dies entspricht einer Pulsperiodendauer von 200 µs. Bei Verwendung der sogenannten "Halbperiodensteuerung" (siehe Abschnitt 4.1) ergibt sich ein Abschnitt von 100 µs. Die berechneten Zeitpunkte innerhalb dieses Abschnitts, welche die Umschaltzeitpunkte des Wechselrichters darstellen, sollen mit einer Ungenauigkeit von höchstens 0.5% ausgegeben werden. Die Taktfrequenz des in Abschnitt 2.2.3 beschriebenen Bausteins muß daher also mindestens

$$f_{\text{grenz}} > 0,005 \frac{1}{100 \mu\text{s}} = 2 \text{ MHz} \quad (3.1)$$

betragen. Für die Anwendung als Stromrichtersteuerung wurde die Einsteckkarte PCI-DAS1602/12 der Fa. ComputerBoards Inc. gewählt, deren Zeitwerke gemäß Datenblatt mit 10 MHz getaktet werden und damit die Bedingung (3.1) erfüllen.

3.4.2.2 Genauigkeit der Datenwandlung

Die digitale Regelung von Antrieben erfordert eine hinreichend genaue Erfassung von Systemgrößen, wie z.B. den Motorströmen. Deshalb müssen analoge Signale zuerst in digitale Daten umgewandelt werden. Dabei ist zu beachten, daß mit geringerer Amplitude der Meßgröße der relative Meßfehler (bezogen auf die Amplitude der Meßgröße) ständig zunimmt, da die Ungenauigkeit der Wandlung (± 1 Bit) konstant bleibt. Trotzdem sollte aber die Meßgenauigkeit auch im Bereich kleiner Meßwerte genügend hoch sein. Da eine immer höhere Auflösung der A/D-Wandlung mit stark ansteigenden Kosten verbunden ist, wird für die vorliegende Asynchronmaschine am Beispiel der Motorströme ein Kriterium für die Mindestauflösung angegeben.

Der Effektivwert des Leerlaufstroms der vorliegenden Antriebsmaschine beträgt ca. 6 A. Dieser Wert soll mit einer Genauigkeit von 1% dargestellt werden. Dadurch ergibt sich eine Mindestauflösung I_δ von 0.06 A. Der Bereich zulässiger Ströme wird auf den Anlaufstrom begrenzt, der auf den fünffachen Betrag des Nennstroms abgeschätzt wird, d.h. $I_{max} = 192.5$ A. Daraus ergibt sich für die notwendige Bitzahl n

$$2^n = \frac{\text{max. Eingangsbereich}}{\text{Mindestauflösung}} = \frac{\Delta I_{max} \sqrt{2}}{I_\delta \sqrt{2}} \quad (3.2)$$

und damit

$$n = \log_2 \left(\frac{\Delta I_{max}}{I_\delta} \right) = \log_2 \left(\frac{385 \text{ A}}{0.06 \text{ A}} \right) = 12.65 \quad (3.3)$$

Der A/D-Wandler muß demnach mindestens eine Auflösung von $n = 13$ Bit aufweisen, um den geforderten Ansprüchen gerecht zu werden.

Als zweites Kriterium gilt die höchstmögliche Wandelrate w eines A/D-Wandlers. Auf Einsteckkarten werden sie häufig mit einem Multiplexer verschaltet, so daß bei k angeschlossenen Kanälen jeder davon mit einer Rate von w/k gewandelt wird. Da nicht alle Kanäle gleichzeitig gewandelt werden können, ist eine hohe Wandelrate der Karte (=Summenabtastrate) notwendig, um den Zeitverzug bedingt durch die serielle Wandlung so gering wie möglich zu halten. Es gilt jedoch auch hier, daß eine schnellere A/D-Wandlung mit stark steigenden Preisen verbunden ist.

Zur Erfassung von analogen Signalen wird die Einsteckkarte CIO-DAS1601/12 der Fa. ComputerBoards Inc. gewählt. Sie bietet eine Summenabtastrate von 160 kHz und eine Auflösung von 12 Bit. Es wäre wie oben beschrieben eigentlich eine Auflösung von mindestens 13 Bit erforderlich, allerdings ist die nächsthöhere erhältliche Auflösung bereits 16 Bit, was mit höheren Anschaffungskosten und kleinerer Summenabtastrate verbunden wäre. Der oben erwähnte Zeitverzug liegt bei dieser Karte bei ca. 4 µs, sodaß sich unzulässig große Verzögerungen bei der Aufnahme der Meßwerte ergäben.

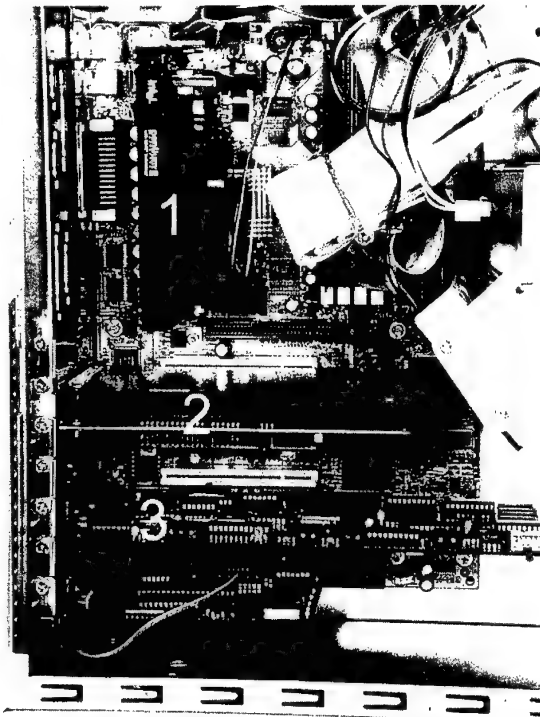


Bild 3.5: Innenansicht des verwendeten AT-Rechners (1 = CPU, 2 = Karte mit Zeitwerk, 3 = Karte mit A/D-Wandler)

Bild 3.5 zeigt die Innenansicht des benutzten PC mit den bereits installierten Einsteckkarten.

3.4.3 Betriebssystem und Hochsprache

3.4.3.1 Betriebssystem

Für Stromrichterbetrieb eignet sich das Betriebssystem DOS im Real Mode am besten, da es auf einfache Weise Zugriffe auf die Systemhardware (z.B. Interruptcontroller) zuläßt. In diesem Betriebsmodus fällt der geringste Systemverwaltungsaufwand für den Prozessor und das Betriebssystem an, da zu jedem Zeitpunkt nur eine Anwendung geöffnet ist (vgl. Abschnitt

2.4.1). Dadurch können Hardwareinterrupts, wie zum Beispiel der "Weckruf" für eine Phasenumschaltung, unter DOS zu genau definierten Zeitpunkten ausgeführt werden.

Da DOS im Real Mode 1 MB Arbeitsspeicher adressieren kann, ist für die Erstellung der Software und für die im Betrieb anfallenden Daten genügend Speicherplatz vorhanden. Off-line Simulationen benötigen allerdings häufig mehr Speicher, der jedoch durch Umschaltung des PC in den Protected Mode bereitgestellt werden kann.

3.4.3.2 Programmiersprache

Prinzipiell kann jede Hochsprache verwendet werden, die den I/O-Adreßbereich ansprechen kann und die Programmierung von Interrupts unterstützt. Außerdem sollte die Hochsprache die Eigenschaft der Modularität besitzen, so daß Teile der Software auch von anderen Anwendungen genutzt werden kann. Aus diesem Grund sind nur Programmiersprachen sinnvoll, die den Binärcode nicht erst zur Laufzeit erzeugen. Dies sind compilerstrukturierte Sprachen wie z.B. Pascal oder C. Interpretergesteuerte Sprachen wie Basic generieren den Binärcode erst zur Laufzeit, und benötigen daher sehr viel mehr Zeit für die Bearbeitung einer Aufgabe. In dieser Arbeit wurde die komplette Software in der Programmiersprache C erstellt. Sie ist nicht hochabstrahierend und eignet sich sehr gut zur Programmierung von Betriebssystemen und Betriebssystemeingriffen [3.8]. Für hardwarenahe Operationen (z.B. bitweises "AND" oder bitweise Rotation) existieren gesonderte Operatoren, sodaß solche Befehle sehr effektiv umgesetzt werden können.

4 Grundlagen von Pulsverfahren

Dieser Abschnitt dient der allgemeinen Darstellung und der im weiteren benutzten Begriffsbestimmung von Pulsverfahren und insbesondere des Prinzips der Halbperiodensteuerung. Die Beschreibung beschränkt sich hier auf Schaltungen mit eingepprägter Gleichspannung. Dabei werden aus einer (meist annähernd konstanten) Eingangsgleichspannung Spannungspulse auf die Last geschaltet. Auf diese Weise können beliebige gewünschte Verläufe der Ausgangsspannung nachgebildet werden.

Die zur Beschreibung von dreiphasigen Pulsverfahren notwendigen Größen und Darstellungsformen werden ebenfalls vorgestellt. Abschließend werden Pulsverfahren zur Erzeugung von Drehstromsystemen mittels B6-Brücken beschrieben.

4.1 Gleichstromsteller

Die einfachste Schaltung, auf die Pulsverfahren angewandt werden können, ist der Tiefsetz-Gleichstromsteller, dessen Prinzip in Bild 4.1 dargestellt ist. Die Eingangsgleichspannung U_d wird zunächst als zeitinvariant angenommen. Neben dem Widerstand R_a enthält die Last noch die Glättungsinduktivität L_a , so daß ein kontinuierlicher Laststrom fließen kann. Beide Elemente werden als konstant angenommen. In vielen Fällen, z.B. in der Antriebstechnik, wirkt lastseitig eine Gegenspannung $e_a(t)$, deren zeitlicher Verlauf im Prinzip beliebig sein kann. Der eigentliche Tiefsetz-Gleichstromsteller besteht aus dem Schalter T1 und der Freilaufdiode D2.

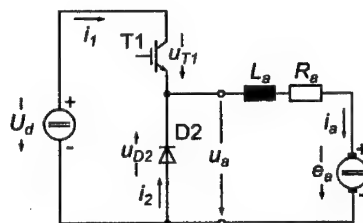


Bild 4.1: Tiefsetz-Gleichstromsteller mit e-R-L-Last

Bei kontinuierlichem Laststrom $i_a(t)$ kann die Ausgangsspannung $u_a(t)$ des Tiefsetzstellers nur zwei Zustände einnehmen. Dies sind

$$u_a = U_d - u_{T1} \quad \text{bei eingeschaltetem Schalter T1 und}$$

$$u_a = 0 - u_{D2} \quad \text{bei ausgeschaltetem Schalter T1.}$$

Die Flußspannungen u_{T1} und u_{D2} werden bei den folgenden prinzipiellen Betrachtungen vernachlässigt.

Bild 4.2 zeigt die Spannungs- und Stromverläufe des Tiefsetzstellers aus Bild 4.1. Die analoge Führungsspannung u_w würde dabei den Laststrom i_w erzeugen. Im Pulsbetrieb wird der Verlauf der Führungsgröße u_w in Pulsperioden der Dauer T_p unterteilt, die sich wiederum aus je einer Halbperiode des Typs "A" (T1 schaltet ein) und des Typs "B" (T1 schaltet aus) zusammensetzen. Die Dauer der Halbperioden wird hier als gleich lang angenommen ($T_{pA} = T_{pB} = T_p/2$). Die Umschaltzeitpunkte t_{0a} , t_{1a} , t_{2a} , usw. können dabei individu-

ell festgelegt werden. Dieses Verfahren wird im folgenden "Halbperiodensteuerung" genannt¹. Bei Lasten, die eine Gegenspannung als wesentliches Element enthalten - also ebenfalls bei Antrieben -, ist es besonders günstig die Mittelwerte von u_w und u_a in jeder Halbperiode gleich groß zu machen, d.h. die beieinander liegenden, gleichartig schattierten Flächen in Bild 4.2 gleich groß. Zur Bestimmung der Umschaltzeitpunkte werden verschiedene Methoden verwendet.

Ein einfacher Weg ist die Abtastung der Führungsgröße. Dabei wird beispielsweise zu Beginn jeder Halbperiode der Abtastwert gespeichert ("Sample and Hold") und danach mit einem dreieckförmigen Hilfssignal u_H verglichen. Die Zeitpunkte, an denen sich beide Signale schneiden, geben die Umschaltzeitpunkte (t_{0a} , t_{1a} , t_{2a} , usw.) an. Bei diesem Verfahren können ggf. größere Abweichungen des Mittelwertes gegenüber der Führungsgröße auftreten. Dieses Verfahren ist aus der Literatur unter den Begriffen "Asymmetric Regular Sampling" [4.1] oder "Double Edge Sampling" [4.2] bekannt. Abtastverfahren wie dieses eignen sich sehr gut für analoge Realisierung der Pulsweiten-Modulation (PWM). Obwohl sie auch auf Digitalrechnern einfach zu realisieren sind, werden sie in dieser Arbeit nicht näher behandelt.

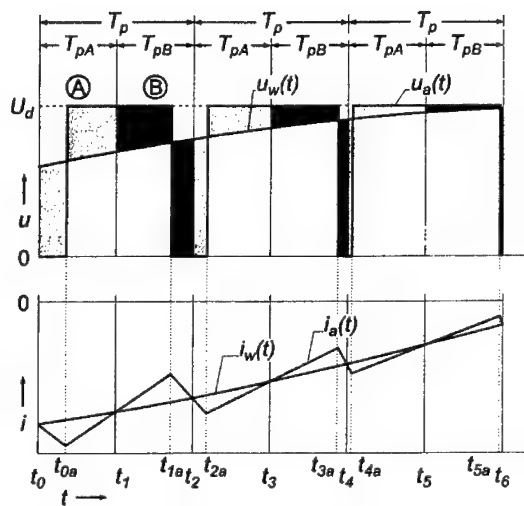


Bild 4.2: Verläufe der Spannungen $u_w(t)$ und $u_a(t)$, sowie der Ströme $i_w(t)$ und $i_a(t)$ eines Tiefsetzstellers bei kontinuierlichem Laststrom

¹Im Gegensatz dazu legt die "Vollperiodensteuerung" immer zwei aufeinander folgende Umschaltungen fest und setzt den Spannungspuls mittig zur Pulsperiode.

Eine andere Möglichkeit besteht in der Berechnung von Mittelwerten der Führungsgröße (siehe Bild 4.3). Dabei wird von einem bekannten, analytisch beschriebenen Verlauf der Führungsgröße u_w ausgegangen, der Mittelwert innerhalb einer Halbperiode berechnet und in Spannungspulse der Eingangsgleichspannung U_d umgesetzt. Man kann dieses Verfahren wie in

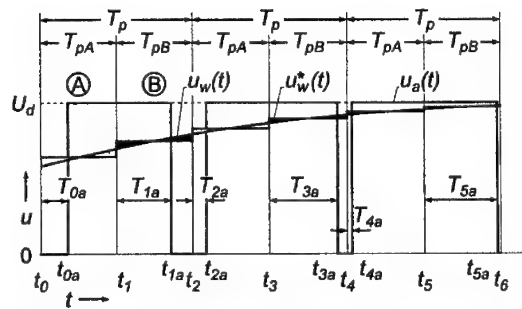


Bild 4.3: Annäherung der Führungsfunktion $u_w(t)$ durch Mittelwertbildung in jeder Halbperiode

Bild 4.3 dadurch beschreiben, daß der Verlauf von u_w zunächst durch eine Stufenfunktion u_w^* mit gleichem Mittelwert in der Halbperiode angenähert wird. Die Höhe der Stufe in der Halbperiode i wird durch die Gleichung

$$u_{wi}^* = \frac{1}{t_{i+1} - t_i} \int_{t_i}^{t_{i+1}} u_w(t) dt \quad (4.1)$$

beschrieben. Wie in Bild 4.3 ersichtlich, sind die grau schattierten Flächen, die durch u_w und u_w^* eingeschlossen werden, gleich. Wenn die gepulste Spannung u_a wieder diesen Mittelwert haben soll, gelten für die Umschaltzeitpunkte der jeweiligen Halbperiode folgende lineare Zusammenhänge.

$$t_{ia} = \begin{cases} T_{pA} \left(1 - \frac{u_{wi}^*}{U_d} \right) & \text{(einschalten, Typ "A", i gerade)} \\ T_{pB} \frac{u_{wi}^*}{U_d} & \text{(ausschalten, Typ "B", i ungerade)} \end{cases} \quad (4.2)$$

Das Verfahren bedeutet, daß bei analoger Realisierung spätestens zum Umschaltzeitpunkt, und bei der unten beschriebenen digitalen Steuerung zu Beginn der Halbperiode oder noch früher, die Stufenspannung u_{wi}^* bekannt sein muß oder Annahmen darüber getroffen sind. Es müssen also vorausberechnende (oder "prädiktive") Verfahren angewandt werden, die sich jedoch gut auf Mikrocomputern realisieren lassen. Insbesondere für Wechselstromanwendungen (siehe Abschnitte 4.2 und 4.3) sind solche Verfahren geeignet und werden deswegen in dieser Arbeit benutzt.

Bei hohen Pulsfrequenzen $f_p = 1/T_p$, d.h. $f_p \gg f_l$, kann auch der Mittelwert der Funktionswerte der Führungsgröße an den Bereichsgrenzen herangezogen werden und berechnet sich so zu

$$u_{wl}^* \approx \frac{u_{wl} + u_{w(i+1)}}{2} \quad (4.3)$$

Der dadurch entstandene Fehler gegenüber dem exakten Mittelwert ist dabei vernachlässigbar.

Es ist nachweisbar, daß bei genügend hoher Pulsfrequenz - d.h. bei $\omega_p L_a \gg R_a^{-1}$ (Bild 4.1) - der Laststrom $i_a(t)$ an den Grenzen der Halbperioden wieder den Verlauf des gewünschten Laststroms $i_w(t)$ erreicht, wie es in Bild 4.2 gezeichnet ist. Zu diesen Zeitpunkten ist also der Laststrom ohne die sonst vorhandenen Oberschwingungsanteile meßbar.

Die Halbperiodensteuerung ermöglicht vom Prinzip her eine beliebige Variation der Halbperiodendauern T_{pA} und T_{pB} ² und damit auch der Periodendauer von $T_p = T_{pA} + T_{pB}$. Dabei erreicht der Mittelwert über eine Pulsperiode der Ausgangsspannung bei gleicher Pulsperiodendauer den gleichen Wert wie im Fall eines symmetrischen Trägersignals. Das Spektrum der Ausgangsspannung enthält aber weitaus mehr Oberschwingungen und sogar Schwebungen [4.3]. Außerdem kann es zu großen Abweichungen des Mittelwertes des Laststroms vom Mittelwert des Führungsstroms i_w kommen, obwohl beide Größen an den Halbperiodengrenzen gleich sind. Bei der weiteren Beschreibung der Pulsverfahren wird immer $T_{pA} = T_{pB} = T_{pH}$ = konstant angenommen, weil dies zu übersichtlichen Darstellungen führt. Die Berechnungen werden aber pro Halbperiode angegeben und gelten damit auch für beliebige Änderungen von T_{pA} und T_{pB} .

Neben der Halbperiodensteuerung wird in der Praxis auch die Vollperiodensteuerung verwendet, die dann entsteht, wenn die Stufen der Spannung u_w^* die Breite einer ganzen Pulsperiode einnehmen. Die Halbperiodensteuerung ist jedoch diejenige Pulsmethode, welche die Führungsspannung u_w^* mit den geringsten Abweichungen nachbildet.

Die Halbperiodensteuerung bringt aber gegenüber der Vollperiodensteuerung einen erhöhten Rechenaufwand und höhere Anforderungen an die Rechengeschwindigkeit mit sich, da in jeder Hälfte einer Pulsperiode das Umschalten der Leistungshalbleiter und ggf. auch Regelungs-routinen durchlaufen werden müssen. Für die angestrebte Mindestpulsfrequenz von 5 kHz ergibt sich die Halbperiodendauer und damit der Berechnungszyklus für die Pulsung zu 100µs.

¹Diese Bedingung ist bei Antrieben mit gepulsten Transistorwechselrichtern fast immer gegeben.

²Dies kann sogar soweit führen, daß eine der beiden Halbperioden mit der Dauer 0 gewählt wird. Dies entspricht dann der Modulation mit einer Sägezahnspannung.

4.2 Phasenschalter

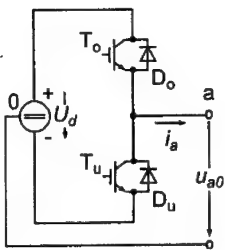


Bild 4.4:
Schaltplan eines Phasenschalters

Für den Aufbau von Stromrichtern, die auch negative Ströme und Spannungen erzeugen können, werden sogenannte "Phasenschalter" (Zweigpaare mit Rücklaufdioden) entsprechend Bild 4.4 verwendet. Für die gebräuchlichste Schaltung zur Erzeugung eines Drehstromsystems (B6-Brücke) sind drei Phasenschalter notwendig.

Die Eingangsgleichspannung U_d kann man fiktiv in zwei Spannungen gleicher Höhe ($\pm U_d/2$) aufteilen. Der dabei entstehende gedachte Nullpunkt "0" ist zugleich Bezugspunkt für die Spannung u_{a0} an der Ausgangsklemme "a". Der Phasenschalter selbst besteht aus zwei antiparallelen Transistor-Dioden-Kombinationen T_o - D_o und T_u - D_u , wovon eine (T_o - D_o) mit dem Potential $+U_d/2$ und die andere mit dem Potential $-U_d/2$ verbunden ist. In gepulsten Systemen werden die Transistoren im wechselseitigen Ausschluß geschaltet, um einen direkten Kurzschluß über T_o und T_u zu vermeiden. Das heißt, wenn T_o leitet, dann sperrt T_u und umgekehrt. Die Klemme "a" liegt somit entweder auf dem Potential $+U_d/2$ oder $-U_d/2$, wodurch der Spannungshub bei einer Umschaltung $\pm U_d$ beträgt. Die Umschaltungen selbst werden als ideal angenommen. Aufgrund des realen Verhaltens der Schaltelemente ist unabhängig von der Schalttrichtung eine kleine Sicherheitszeit erforderlich, in der beide Transistoren ausgeschaltet sind (siehe Abschnitt 3.3). Das führt zu einer Beeinflussung des realen Umschalt Augenblicks - und damit auch der Höhe der Ausgangsspannung - durch die Richtung des Ausgangsströms i_a im Phasenschalter. Diese Effekte werden hier nicht behandelt, sind aber aus der Literatur (z.B. [4.4]) bekannt. Die dort beschriebenen Gegenmaßnahmen können in der Regelung mit PC ohne besondere Schwierigkeiten eingebaut werden.

Im Grunde müßten auch hier die Flußspannungen u_T und u_D der oberen bzw. unteren Leistungshalbleiter berücksichtigt werden. Im allgemeinen sind sie allerdings sehr klein gegenüber der Zwischenkreisspannung, so daß sie in der vorliegenden Arbeit vernachlässigt werden können.

In vielen Veröffentlichungen wird der Schaltzustand eines Phasenschalters nicht über dessen Ausgangsklemmenspannung, sondern über die diskreten Schaltzustände desjenigen Transistors angegeben, der an das Potential $+U_d/2$ angeschlossen ist. Dabei wird der Zustand

"eingeschaltet" durch "1" und der Zustand "ausgeschaltet" durch "0" repräsentiert. Dann ergibt sich für den Zustand z eines Phasenschalters

$$z = \begin{cases} "0" & T_o \text{ ausgeschaltet, } T_u \text{ eingeschaltet} \\ "1" & T_o \text{ eingeschaltet, } T_u \text{ ausgeschaltet} \end{cases} \quad (4.4)$$

Da in Sonderfällen jedoch auch der Zustand T_o und T_u ausgeschaltet auftritt, wird bei der Realisierung (Abschnitt 5) eine Darstellung benutzt, die den Zustand jeder Transistor-Dioden-Kombination getrennt angibt.

Die Vorgehensweise bezüglich der Halbperiodensteuerung und Mittelwertbildung von u_{wi}^* nach Gleichung (4.1) bzw. (4.3) an sich bleibt jedoch von der Art der Notation unberührt. Durch die Aufteilung der Eingangsspannung in $\pm U_d/2$ ergeben sich allerdings Änderungen bei der Berechnung des Umschaltzeitpunkts in der Halbperiode i .

$$t_{ia} = T_{pH} \left(\frac{1}{2} \mp \frac{u_{wi}^*}{U_d} \right), \quad \begin{array}{l} \text{"-"} \text{ für Typ "A"} \\ \text{"+" für Typ "B"} \end{array} \quad (4.5)$$

Pulsverfahren lassen sich durch die Abfolge des Verhältnisses von Einschaltdauer T_E zu Halbperiodendauer T_{pH} , das sogenannte Einschaltverhältnis $\lambda_o = T_E/T_{pH}$, eindeutig beschreiben. Unter Berücksichtigung der Halbperiodensteuerung ergeben sich für den oberen Schalter folgende vom Typ der Halbperiode abhängige Ausdrücke:

$$\lambda_o = \begin{cases} \frac{T_{pA} - t_{ia}}{T_{pA}}, & \text{Typ A, Einschalten +} \\ \frac{t_{ia}}{T_{pB}}, & \text{Typ B, Einschalten -} \end{cases} \quad (4.6)$$

Daraus kann das Einschaltverhältnis λ_u des unteren Schalters abgeleitet werden. Bei idealen Umschaltungen beträgt es stets $\lambda_u = 1 - \lambda_o$.

Neben diesem Phasenschalter mit zwei Spannungszuständen (bei regulärem Betrieb) werden auch "Drei- oder Mehrpunktschaltungen" verwendet, die entsprechend mehr Ausgangszustände annehmen können, aber auch wesentlich komplexer im Aufbau sind. Solche Schaltungen werden im folgenden nicht behandelt. Die Steuerung mit dem PC ist hier nur bei wesentlich kleineren Pulsfrequenzen sinnvoll.

4.3 Dreiphasige Pulsverfahren

4.3.1 Stromrichterschaltung B6

Die gebräuchlichste Schaltung zur Erzeugung von Drehstromsystemen ist die B6-Brücke¹ entsprechend Bild 4.5, die aus drei Phasenschaltern (A, B, C) besteht. Die eingezeichnete Nummernfolge 1 bis 6 der Transistor-Dioden-Kombinationen entspricht dabei der Einschaltfolge bei Vollaussteuerung. Der gedachte Nullpunkt "0" wird auch hier als Bezugspunkt für die Ausgangsspannungen u_{a0} , u_{b0} , u_{c0} sowie für die Spannung u_{M0} des Mittelpunkts gegenüber "0" herangezogen. Die Last wird weiter unten besprochen. Hier wird nur darauf hingewiesen, daß die Spannung des realen Mittelpunkts M einer im Stern geschalteten Last fast immer von der des Punktes 0 abweicht.

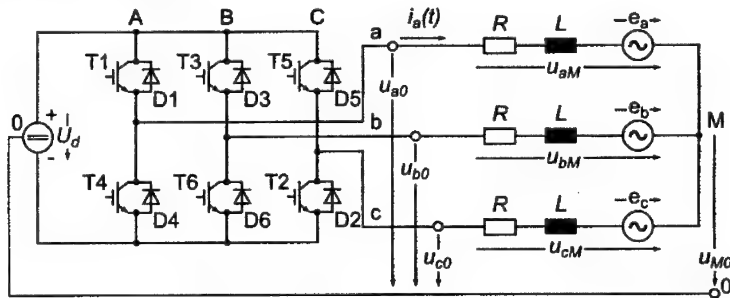


Bild 4.5: Dreiphasiger Stromrichter mit Last in Sternschaltung

4.3.2 Symmetrische dreiphasige Systeme

Ein dreiphasiges, symmetrisches System, bestehend aus drei gegeneinander um 120° phasenverschobenen Größen gleicher Frequenz ω und Amplitude \hat{u} läßt sich wie folgt beschreiben:

$$\begin{aligned} u_a &= \hat{u} \cos(\omega t) \\ u_b &= \hat{u} \cos(\omega t - 2\pi/3) \\ u_c &= \hat{u} \cos(\omega t - 4\pi/3) \end{aligned} \quad (4.7)$$

und damit gilt auch

$$u_a(t) + u_b(t) + u_c(t) = 3 u_0(t) = 0 \quad (4.8)$$

¹Andere Schaltungen bestehen teilweise nur aus zwei Phasenschaltern oder mehr als drei. Sie sind jedoch fast nur in Nischenanwendungen zu finden.

Es ist dabei gleichgültig, ob es ein Stromsystem, ein Spannungssystem oder z.B. ein Flußsystem darstellt. Symmetrische sinusförmige Systeme enthalten also keine Nullkomponente $u_0(t)$ in den Augenblickswerten.

4.3.3 Nullkomponente der Augenblickswerte

Für die Beschreibung von Pulsvorgängen kann eine symmetrische Drehstrommaschine durch ein Modell, bestehend aus einem Widerstand R , einer Induktivität L und einer Gegenspannung pro Phase dargestellt werden, wobei die Spannungen e_a, e_b, e_c mit guter Näherung ein symmetrisches dreiphasiges System darstellen. Dies gilt auch für den Zeitbereich einer Pulsperiode auch für Kurzschlußläufermaschinen, da sich der Hauptfluß - und damit die EMK - wegen des Läuferkurzschlusses nur langsam ändert. Die Wicklungswiderstände kann man für Pulsvorgänge vernachlässigen (siehe auch Abschnitt 4.1). In Bild 4.5 ist ein derartiges Modell in Sternschaltung eingezeichnet. Zur Berechnung der Mittelpunktsspannung u_{M0} lassen sich unter Berücksichtigung der Schaltung im Mittelpunkt M ("freier Sternpunkt") folgende Maschengleichungen aufstellen:

$$\begin{aligned} u_{a0} - u_{M0} &= i_a R + L \frac{di_a}{dt} + e_a \\ u_{b0} - u_{M0} &= i_b R + L \frac{di_b}{dt} + e_b \\ u_{c0} - u_{M0} &= i_c R + L \frac{di_c}{dt} + e_c \end{aligned} \quad (4.9)$$

Da die Summe der Ströme und die Summe der Gegenspannungen Null ist, ergibt sich die Sternpunktspannung zu

$$u_{M0} = \frac{1}{3}(u_{a0} + u_{b0} + u_{c0}) \quad (4.10)$$

Diese Spannung wird auch Nullkomponente der Augenblickswerte genannt. Sie tritt immer im Mittelpunkt von solchen Lasten auf, die aus Pulswechselrichtern gespeist werden. Durch die in jeder Phase einstellbaren äußeren Spannungszustände von $\pm U_d/2$ wird im regulären Pulsbetrieb die Summe der Momentanwerte aller drei Phasenspannungen entsprechend Abschnitt 4.3.4.1 nie Null (z.B. $u_{M0} = 1/3 (+U_d/2 + U_d/2 - U_d/2) = 1/6 U_d \neq 0$). Wegen des freien Sternpunkts gilt im Mittelpunkt stets $i_a + i_b + i_c = 0$. Damit wird die Nullkomponente des Spannungssystems wirkungslos bezüglich der Ströme, wodurch die durch den Wechselrichter hervorgerufenen Oberschwingungsströme reduziert werden. Symmetrische Lasten in Dreieckschaltung unterdrücken bereits von sich aus eine von außen eingeprägte Nullkomponente.

Es ist aber auch möglich, die nachzubildenden Sollwertspannungen u_{wa}, u_{wb} und u_{wc} mit einer Nullkomponente zu versehen. Dies wirkt sich auf die Lage und Länge des Spannungspulses

der einzelnen Phasen aus. Der Mittelwert der Spannungen u_{aM} , u_{bM} , u_{cM} über eine Pulsperiode bleibt davon aber unberührt. Dadurch lassen sich besondere Effekte erzielen, die weiter unten beschrieben werden.

4.3.4 Raumzeiger

Raumzeiger sind heute das übliche Mittel zur Darstellung von Effekten in dreiphasigen Systemen. Durch die Pulsverfahren ergeben sich zusätzliche Besonderheiten, die nicht allgemein bekannt sind. Deswegen wird hier noch einmal kurz auf die Grundlagen der Raumzeigertheorie eingegangen.

Die Beschreibung von dreiphasigen Systemen in der Zeitebene ist mit relativ großem Aufwand verbunden, da für jede Phase deren Verlauf angegeben werden muß. Mit Hilfe einer Drehtransformationsmatrix kann man die Stranggrößen u_a , u_b , u_c auf ein System mit den Größen u_α , u_β und u_0 abbilden¹.

$$\begin{pmatrix} u_\alpha(t) \\ u_\beta(t) \\ u_0(t) \end{pmatrix} = \frac{2}{3} \begin{pmatrix} \cos 0 & \cos(2\pi/3) & \cos(4\pi/3) \\ \sin 0 & \sin(2\pi/3) & \sin(4\pi/3) \\ 1/2 & 1/2 & 1/2 \end{pmatrix} \begin{pmatrix} u_a(t) \\ u_b(t) \\ u_c(t) \end{pmatrix} \quad (4.11)$$

Die zwei Größen u_α , u_β bilden die beiden Komponenten des sogenannten Raumzeigers. Die Größe u_0 ist die bereits bekannte Nullkomponente. Die Rücktransformation vom α , β , 0-System in reale Stranggrößen erfolgt mit

$$\begin{pmatrix} u_a \\ u_b \\ u_c \end{pmatrix} = \begin{pmatrix} \cos 0 & \sin 0 & 1 \\ \cos(2\pi/3) & \sin(2\pi/3) & 1 \\ \cos(4\pi/3) & \sin(4\pi/3) & 1 \end{pmatrix} \begin{pmatrix} u_\alpha \\ u_\beta \\ u_0 \end{pmatrix} \quad (4.12)$$

Entfällt nun beispielsweise durch Schaltungszwang die Nullkomponente, dann können Gleichung (4.11) und (4.12) noch weiter zu

$$\begin{pmatrix} u_\alpha(t) \\ u_\beta(t) \end{pmatrix} = \frac{2}{3} \begin{pmatrix} \cos 0 & \cos(2\pi/3) & \cos(4\pi/3) \\ \sin 0 & \sin(2\pi/3) & \sin(4\pi/3) \end{pmatrix} \begin{pmatrix} u_a(t) \\ u_b(t) \\ u_c(t) \end{pmatrix} \quad (4.13 \text{ a})$$

bzw.

¹Gleichung (4.11) gibt die in der Literatur am häufigsten verwendete Definition wieder, die eine besonders einfache Rücktransformation ermöglicht. Es werden aber auch andere Faktoren als $2/3$ verwendet.

$$\begin{pmatrix} u_a(t) \\ u_b(t) \\ u_c(t) \end{pmatrix} = \begin{pmatrix} \cos 0 & \sin 0 \\ \cos(2\pi/3) & \sin(2\pi/3) \\ \cos(4\pi/3) & \sin(4\pi/3) \end{pmatrix} \begin{pmatrix} u_a(t) \\ u_\beta(t) \end{pmatrix} \quad (4.13 \text{ b})$$

vereinfacht werden. Die transformierte Größe besteht dann nur noch aus zwei voneinander linear unabhängigen Komponenten, die als komplexe Größe $\underline{u}(t)$ darstellbar sind.

$$\underline{u}(t) = u_a + j u_\beta = \frac{2}{3} \left(u_a(t) + \underline{a} u_b(t) + \underline{a}^2 u_c(t) \right), \quad \underline{a} = e^{j2\pi/3} \quad (4.14)$$

Der komplexe Raumzeiger $\underline{u}(t)$ ist eine anschauliche Darstellung von Drehstromsystemen unter Ausschaltung der Nullkomponente. Damit wird die Zahl der Komponenten von drei auf zwei reduziert. Für die Rücktransformation gilt

$$\begin{aligned} u_a &= \operatorname{Re}(\underline{u}) \\ u_b &= \operatorname{Re}(\underline{a}^2 \underline{u}) \\ u_c &= \operatorname{Re}(\underline{a} \underline{u}) \end{aligned} \quad (4.15)$$

angegeben werden. Sie liefert allerdings nur die Phasengrößen ohne Berücksichtigung der Nullkomponente. Die Raumzeigerdarstellung kann also auch dazu verwendet werden, um aus den Phasengrößen u_{a0}, u_{b0}, u_{c0} mit Nullkomponente die Ströme i_a, i_b und i_c (die keine Nullkomponente enthalten) zu berechnen. Im Ursprungssystem (u_{a0}, u_{b0}, u_{c0}) ist jedoch immer die Darstellung der drei Stranggrößen und der Nullkomponente erforderlich, um das System exakt zu beschreiben.

Mit Hilfe von Raumzeigern können auch unsymmetrische und zeitdiskrete äußere Spannungen und Ströme dargestellt werden. Die Behandlung mit Raumzeigern umfaßt dabei vor allem Überlegungen des fehlerbehafteten Betriebs von Netzen und elektrischen Maschinen (z.B. ein- oder mehrphasige Kurzschlüsse), die in der Literatur schon mehrfach beschrieben wurden ([4.5], [4.6], [4.7]). Stromrichter stellen zeitdiskrete Systeme dar. Wie im Anschluß gezeigt wird, führt die Verwendung von Raumzeigern bei Stromrichtern zu einer anschaulichen Beschreibungsweise. Durch ihre Übertragbarkeit auf weitere elektrische (z.B. Oberschwingungen, Strombeläge) und magnetische (z.B. Flüsse, Induktionen usw.) Größen sowie durch deren universelle Einsatzmöglichkeiten wurden Raumzeiger zu einem wichtigen Werkzeug in der elektrischen Energie- und Antriebstechnik.

4.3.4.1 Diagramm der Stromrichter-Spannungszustände

In Schaltungen nach Bild 4.5 kann jede der drei Phasen unabhängig voneinander die diskreten Spannungszustände $+U_d/2$ und $-U_d/2$ einnehmen. Auf diese Weise ergeben sich am Stromrichter Ausgang acht verschiedene Zustände, für welche die Leiterspannungen u_{a0}, u_{b0}, u_{c0} , die

4.3.4.2 Stromraumzeiger

Im folgenden sollen die Auswirkungen des Pulsbetriebs auf die Oberschwingungsströme der Last näher betrachtet werden. Dabei können die Widerstände R des Ersatzschaltplans von Bild 4.5 vernachlässigt werden, und es ergibt sich der einfache Ersatzschaltplan von Bild 4.7. Die Größen \underline{u} , \underline{u}_L , \underline{e} sowie \underline{i}_L stellen dabei Raumzeiger dar. Die mathematische Umsetzung des Schaltplans darf nicht mit den Methoden der Zeitzeigertheorie erfolgen, sondern führt zu Differentialgleichungen [4.9]. Bild 4.8 zeigt ein Beispiel für die Spannungsraumzeiger. Der Gegenspannungsraumzeiger \underline{e}_w ¹ soll während einer Pulsperiode durch eine geeignete Kombination aus den Spannungsraumzeigern ${}^0\underline{U}$, ${}^1\underline{U}$, ${}^2\underline{U}$ und ${}^7\underline{U}$ nachgebildet werden. Dazu werden die genannten Raumzeiger jeweils eine Zeitdauer 0T , 1T , 2T und 7T eingeschaltet. Während dieser Zeiten steht an der Induktivität entsprechend Gleichung (4.16) der resultierende Spannungsraumzeiger ${}^1\underline{U} - \underline{e}_w$ an. Die Richtung dieses Raumzeigers bestimmt für jeden Zeitabschnitt die Richtung der Änderung des Stromraumzeigers. Aus der Maschengleichung folgt für den Stromraumzeiger \underline{i}_L

$$\underline{i}_L = \underline{i}(0) + \frac{1}{L} \int_0^t (\underline{u}(t) - \underline{e}(t)) dt \quad (4.16)$$

Für das Beispiel von Bild 4.8 ergibt sich für eine Pulsperiode ein Verlauf, wie er in Bild 4.9 gezeichnet ist. Der Raumzeiger $\underline{i}(0)$ ist der Anfangswert an der Halbperiodengrenze. Er liegt entsprechend Bild 4.2 auf dem Verlauf der Sollwertkurve $\underline{i}_{sw}(t)$ und spielt für den relativen Verlauf keine Rolle. $\underline{i}(0)$ wurde in Bild 4.9 unrealistisch klein gezeichnet. Der Einfachheit halber wurde zugrunde gelegt, daß derselbe Sollwertraumzeiger \underline{u}^* in den zwei aufeinanderfolgenden Halbperioden T_{pA} und T_{pB} gilt. In jeder Halbperiode tritt ein dreieckförmiger Verlauf des Stromraumzeigers auf, der hier aus den Anteilen ${}^0\underline{i}$, ${}^1\underline{i}$, ${}^2\underline{i}$ und ${}^7\underline{i}$ besteht. Die Ziffern 0, 1, 2 und 7 bezeichnen dabei den Anteil, der durch das Anlegen des entsprechenden Stromrichterzustandes hervorgerufen wird. Es ist zu beachten, daß durch die Gegenspannung auch während der Einschaltdauer der Nullspannungsraumzeiger ${}^0\underline{U}$ und ${}^7\underline{U}$ eine Stromänderung auftritt.

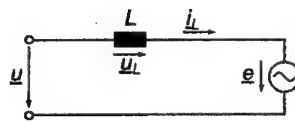


Bild 4.7:
Raumzeiger-Ersatzschaltbild einer dreiphasigen e - L -Last

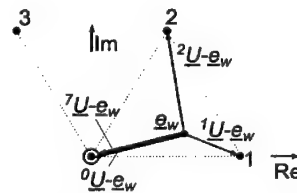


Bild 4.8:
Spannungsraumzeiger-Diagramm mit Gegenspannung

¹Zur Vereinfachung der Darstellung wird der Gegenspannungsraumzeiger \underline{e} für die Pulsperiode als konstant angenommen. Die Bilder gelten also exakt nur für eine stufenförmige Gegenspannung.

Es ist im Bild auch zu erkennen, daß der Stromraumzeiger durch den Pulsbetrieb nicht mehr auf einer Kreiskurve verläuft, wie es bei rein sinusförmigen Strömen der Fall wäre, sondern von den aus dem dreieckförmigen Verlauf hervorgehenden Oberschwingungen überlagert ist. Der Oberschwingungsraumzeiger \vec{i}_h ergibt sich als Abweichung des Gesamtstromraumzeiger \vec{i} vom Grundschwingungsstromraumzeiger \vec{i}_l .

$$\vec{i}_h = \vec{i} - \vec{i}_l \quad (4.17)$$

Für hohe Pulsfrequenzen ($f_p \gg f_l$) sind die Oberschwingungsströme vergleichsweise klein. Sie können jedoch insbesondere bei Vollaussteuerung zu Stromüberhöhungen führen, die gesondert betrachtet werden müssen. Da Oberschwingungsströme in der Last erhöhte Verluste verursachen, ist die Minimierung dieses Stromanteils meist ein Hauptkriterium für die Untersuchung und den Entwurf von Pulsverfahren [4.10], [4.11].

Bild 4.9 zeigt auch, daß an den Halbperiodengrenzen der Oberschwingungsanteil des Stromraumzeigers Null wird, sodaß auch hier - analog zum Gleichstromsteller - diese Zeitpunkte für Strommessungen günstig sind.

4.3.5 Modulationsgrad und Aussteuerungsgrad

Der Modulationsgrad M nach [4.12] bezeichnet das Verhältnis zwischen dem Scheitelwert der Grundschwingung der Ausgangsspannung $\hat{u}_{abc0,l}$ zur halben Zwischenkreisspannung $U_d/2$. Er kann abhängig vom Modulations- (bzw. Puls-)verfahren (siehe Abschnitt 4.3.6) unterschiedliche Höchstwerte annehmen. Der maximale Wert der Aussteuerung wird bei Vollaussteuerung erreicht und beträgt dort

$$M_{max} = \frac{4}{\pi} \quad (4.18)$$

Der Modulationsgrad M ist bei symmetrischer Aussteuerung für alle drei Phasen gleich. Es ist aber auch sinnvoll, für jede Phase einen eigenen, zeitabhängigen Modulationsgrad $m_a(t)$, $m_b(t)$, $m_c(t)$ anzugeben, d.h.

$$m_i = \frac{e_{wi}}{U_d/2}, \quad i = a, b, c \quad (4.19)$$

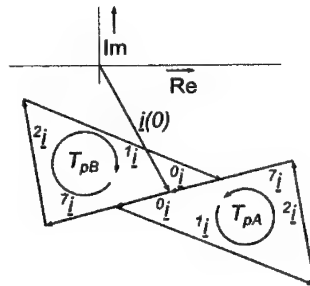


Bild 4.9:
Verlauf des Stromraumzeigers

Für ein Spannungssystem nach Gleichung (4.7 und 4.8) ändern sich die Modulationsgrade sinusförmig und hätten gegeneinander eine Phasenverschiebung von 120° . Die größtmögliche Spannung der Phasen beträgt $U_d/2$, also gilt für die Modulationsgrade

$$-1 \leq m_i \leq +1, \quad i = a, b, c \quad (4.20)$$

In Anlehnung an dreiphasige Systeme von physikalischen Größen kann ein komplexer Modulationsgrad

$$\underline{m}(t) = \frac{2}{3} (m_a(t) + \underline{a} m_b(t) + \underline{a}^2 m_c(t)) = M e^{j\alpha} = m_x + j m_y \quad (4.21)$$

definiert werden, der sich ebenfalls als Raumzeiger in der komplexen Ebene gemäß Bild 4.6 sowohl in polaren wie auch kartesischen Koordinaten darstellen läßt. Die Eckpunkte 1 bis 6 des Raumzeigerbilds 4.6 entsprechen dabei einem Betrag des komplexen Modulationsgrads $|\underline{m}|=4/3$.

Den Aussteuerungsgrad A nach DIN 41750 bezieht man hier sinnvoller Weise auf die Grundschwingung. Er kennzeichnet dann das Verhältnis von aktueller zu maximaler Grundschwingung und ergibt sich zu

$$A = \frac{M}{M_{\max}} = \frac{\pi}{4} M \quad (4.22)$$

4.3.6 Beschreibung verschiedener Pulsverfahren

Das Ziel von dreiphasigen Pulsverfahren ist zunächst die Erzeugung von Drehspannungssystemen, in der Antriebstechnik meist mit variabler Frequenz und Amplitude. Wie bereits erwähnt treten durch den Pulsbetrieb aber zusätzliche (unerwünschte) Effekte auf, wie z.B. erhöhte Oberschwingungen, die sich auch auf die Last auswirken können. Mit Hilfe von speziellen Pulsverfahren können solche Effekte reduziert oder neue, erwünschte Effekte erzeugt werden.

Zur Einteilung und zum Vergleich solcher Verfahren gibt die Literatur verschiedene Ansätze (z.B. [4.13], [4.14]). In dieser Arbeit werden diejenigen Verfahren genauer behandelt, die einen Raumzeiger entsprechend Bild 4.8 nachbilden. Dafür sind die Einschalt Dauern der Raumzeiger 1 und 2 sowie die Summe der Einschalt Dauern 0T und 7T der Nullspannungsraumzeiger für jeden Sollspannungsraumzeiger \underline{e}_{sw} fest vorgegeben. Ein Freiheitsgrad für den Entwurf von Pulsverfahren besteht aber in der Aufteilung der Summe auf die beiden Nullspannungsraumzeiger. Das entspricht der Änderung der Nullkomponente bzw. deren Einführung in die Stufen der Spannungen u_{w00}^* , u_{w01}^* , u_{w02}^* . Solche Pulsverfahren können im Prinzip auf zwei Arten angegeben werden: Man kann sie über analytische, meist abschnittsweise definierte Funktionen beschreiben [4.15]. Eine zweite Möglichkeit besteht darin, eine für das Pulsverfahren allge-

meine Berechnungsvorschrift für die Nullkomponente anzugeben, und diese dann auf die einzelnen Phasen umzulegen [4.16].

Es existieren außerdem noch Pulsverfahren, die einen Raumzeiger \underline{e}_w durch die Verwendung von nur zwei Spannungszuständen nachzubilden versuchen¹. In Bild 4.8 wären das die Zustände ${}^0\underline{U}$ und ${}^1\underline{U}$. Dadurch kann der Raumzeiger jedoch nur grob angenähert werden. Die Folge sind zusätzliche Oberschwingungen. Die sogenannte "Vollaussteuerung" ist ein Sonderfall dieser Art der Pulsverfahren, bei dem die Phasen nur nach jeweils einer halben Grundschwingungsdauer umgeschaltet werden. Solche Verfahren werden meist nur bei kleinen Pulsfrequenzen angewandt und deshalb hier nicht weiter behandelt. Die später beschriebene Realisierung der Steuerung mit dem PC ermöglicht diese Pulsverfahren aber ohne weiteres.

In den folgenden Abschnitten werden jeweils zwei Halbperioden mit gleichem komplexem Modulationsgrad angenommen. Dadurch werden die speziellen Effekte der unterschiedlichen Pulsverfahren besonders deutlich.

4.3.6.1 Sinus-Modulation der Einzelphasen

Vorgegeben sei ein symmetrisches, sinusförmiges Sollwert-Spannungssystem entsprechend Gleichung (4.7). Durch die Transformation nach Gleichung (4.14) wird es auf einen konstant umlaufenden Raumzeiger abgebildet. Da das Ausgangssystem keine Nullkomponente besitzt, würde eine Rücktransformation des Raumzeigers in reale Größen wiederum auf ein symmetrisches, sinusförmiges System führen. Deshalb kann das Ursprungssystem selbst zur Berechnung der Umschaltzeitpunkte benutzt werden. Bei Umwandlung des sinusförmigen Verlaufs in Stufen ergeben sich je nach

Art der Stufenbildung für die Dauer einer halben oder ganzen Pulsperiode konstante Raumzeiger. Bild 4.10 zeigt den Ausschnitt aus einem solchen System, das genau den Raumzeiger \underline{e}_w bilden würde, wie er in Bild 4.8 gezeichnet ist. Unter Benutzung von zeitabhängigen Modulationsgra-

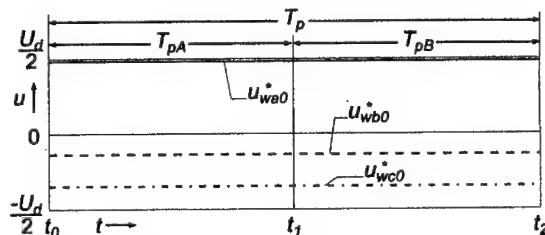


Bild 4.10: Ausschnitt eines symmetrischen, sinusförmigen Sollwert-Spannungssystems (Mittelwerte). Dieses Bild zeigt die Sollwerte für a) in Bild 4.12.

¹Solche Pulsverfahren schalten pro Halbperiode höchstens eine Phase. Deshalb können sie auch als "einphasige Pulsverfahren" bezeichnet werden.

den m_a , m_b und m_c für die zugehörigen Phasen nach Gleichung (4.19) ergeben sich die Umschaltzeitpunkte bei Halbperiodensteuerung zu

$$\left. \begin{aligned} t_{0i} &= \frac{T_{pA}}{2} (1 - m_i), & \text{Einschalten } +, \text{ Typ A} \\ t_{1i} &= \frac{T_{pB}}{2} (1 + m_i), & \text{Einschalten } -, \text{ Typ B} \end{aligned} \right\} i = a, b, c \quad (4.23)$$

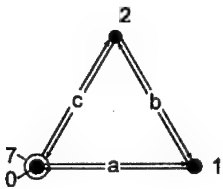


Bild 4.11:
Durchlauf der Zustände
0, 1, 2, 7

Bild 4.12 zeigt unter a) den resultierenden Verlauf der Phasenspannungen u_{a0} , u_{b0} und u_{c0} , der sich aus den in Bild 4.10 vorgegebenen Sollwerten einstellt. Die Reihenfolge der Umschaltungen in der Pulsperiode von t_0 bis t_2 lautet a-b-c-c-b-a. Dies läßt sich auch durch die Reihenfolge der Stromrichterzustände ausdrücken und ergibt dann 0-1-2-7-2-1-0 entsprechend Bild 4.11. Da die Sollwerte den Raumzeiger \underline{z}_w aus Bild 4.8 bilden und keine Nullkomponente enthalten, ist auch der resultierende Stromraumzeiger mit dem in Bild 4.9 identisch.

Bei diesem Pulsverfahren ist die Aussteuerung dann erreicht, wenn ein Umschaltzeitpunkt den Wert 0 oder T_{pA} bzw. T_{pB} annimmt. Das führt zu Scheitelwerten der Ausgangsspannung die kleiner oder gleich $U_d/2$ sind, so daß der Modulationsgrad M maximal den Wert 1 bzw. der Aussteuerungsgrad A den Wert 0,785 erreicht.

4.3.6.2 Dreiphasiges Pulsen mit symmetrischen Nullzuständen

In a) von Bild 4.12 ist deutlich zu erkennen, daß die Nullspannungszustände 0 und 7 unterschiedlich lang eingeschaltet sind (Zeiten 0T_A und 7T_A). Wie bereits erwähnt kann in das Spannungssystem eine Nullkomponente eingeführt werden, die im Mittelwert des Stromsystems wirkungslos ist. Sie kann so gestaltet werden, daß die beiden Nullspannungszustände 0 und 7 in jeder Halbperiode symmetrisch liegen, d.h. ${}^0T = {}^7T$ wird. Dadurch kommt man zum sogenannten symmetrischen dreiphasigen Pulsen wie es in b) von Bild 4.12 gezeichnet ist. Es ist zu erkennen, daß sich nur die Einschalt Dauern 0T und 7T verändern, während die Dauern 1T und 2T unverändert bleiben. Dies entspricht einer Verschiebung um T_{Null} der Umschaltzeitpunkte. Bild 4.12 b) zeigt auch, daß abhängig vom Typ der Halbperiode die Zeitdauern bis zu den Phasenumschaltungen um die Zeitdauer T_{Null} verringert oder vergrößert werden. Da während einer Periode der Grundschwingung die zuerst zu schaltende Phase wechselt, berechnet sich T_{Null} allgemein zu

$$T_{Null} = \frac{T_{pA,B} - \max(t_{0a}, t_{0b}, t_{0c}) - \min(t_{0a}, t_{0b}, t_{0c})}{2}, \text{ Typen A und B} \quad (4.24)$$

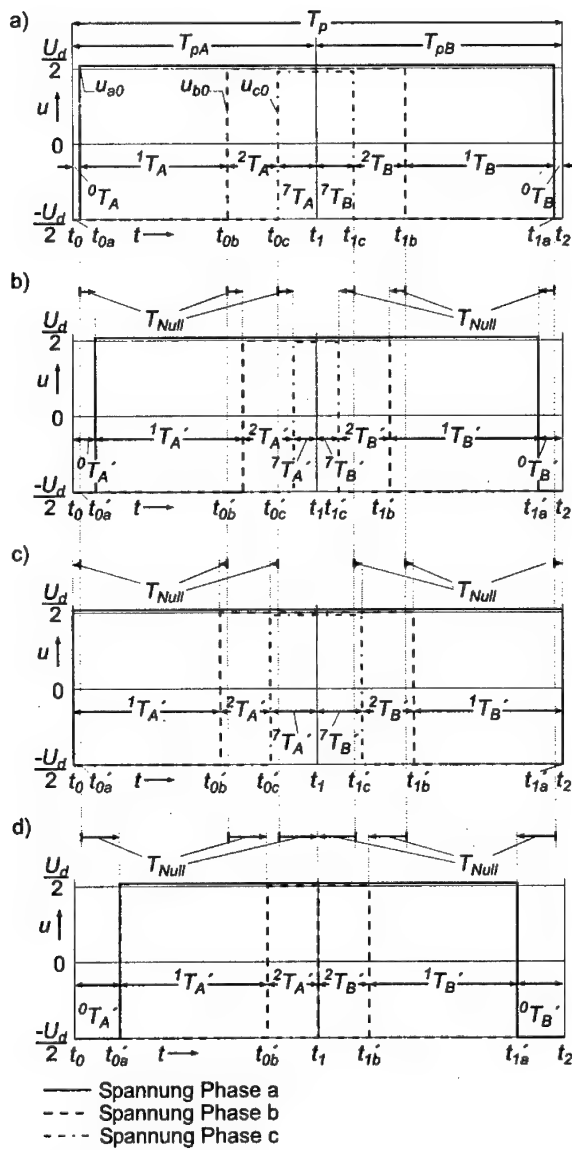


Bild 4.12: Verläufe der Phasenspannungen während einer Pulsperiode für
 a) Sinus-Modulation der Einzelphasen
 b) symmetrische Nullspannungszustände
 c) Zweiphasiges Pulsen erster Art
 d) Zweiphasiges Pulsen zweiter Art

Die neuen Umschaltzeitpunkte lauten dann also

$$t_i' = t_i + T_{Null}, \quad i = a, b, c \quad (4.25)$$

Durch die Symmetrierung der Nullzustände ergeben sich auch gleichlange Stromraumzeiger-Anteile 0_i und 7_i . Dadurch verschwindet bei gleichem Modulationsgrad für beide Halbperioden der in Bild 4.9 deutlich zu sehende Versatz zweier zu einer Pulsperiode gehörender Dreiecke, so daß sich ein Parallelogramm entsprechend Bild 4.13 bildet. Das führt zu deutlich reduzierten Oberschwingungen (siehe [4.10]). Deswegen wird dieses Verfahren bevorzugt eingesetzt.

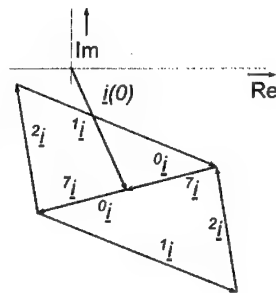


Bild 4.14:
Stromraumzeiger mit symmetrischen Nullzuständen

Die symmetrische Aufteilung der Nullzustände bewirkt, daß bei Erhöhung der Amplitude des Modulationsgrads m die Einschaltauern der beiden Nullspannungsraumzeiger gleichzeitig Null werden. Damit kann die Summe aus $1T$ und $2T$ maximal die Halbperiodendauer erreichen. Bei beliebiger Aufteilung dieser beiden Dauern können also alle Punkte im Inneren des durch die Stromrichterzustände 1 bis 6 aufgespannten Sechsecks erreicht werden. Ein auf einer Kreiskurve umlaufender Raumzeiger kann so maximal den Inkreis des Sechsecks beschreiben entsprechend Bild 4.14. Dadurch können höhere maximale Modulations- und Aussteuerungsgrade ($\sqrt{2}/3$ bzw. 0.907) erzielt werden als bei der Sinusmodulation der Einzelphasen (1 bzw. 0.785), deren maximale Grundschwingung in Bild 4.14 jeweils als Kreis eingezeichnet ist. Zum Vergleich ist auch die Grundschwingung bei Vollaussteuerung ($A=1$) eingezeichnet.

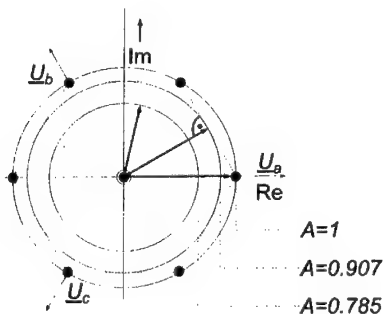


Bild 4.14:
Realisierbare Aussteuerungsgrade für Sinusmodulation ($A=0.785$), für Pulsen mit symmetrischen Nullzuständen und für Vollaussteuerung ($A=1$)

4.3.6.3 Zweiphasiges Pulsen erster Art

Die Aufteilung der Einschaltdauer der Nullzustände $0T$ und $7T$ kann auch so erfolgen, daß der kürzer eingeschaltete Nullzustand überhaupt nicht mehr auftritt. Das kann in der Weise interpretiert werden, daß jeweils die dem Betrag nach größte Aussteuerung der drei Phasen in einer Halbperiode den Wert ± 1 annimmt. Im Gegensatz zum symmetrischen dreiphasigen Pulsen muß deshalb für die Berechnung einer der Nullkomponente entsprechenden zeitlichen Verschiebung T_{Null} der Umschaltzeitpunkte eine Fallunterscheidung hinsichtlich der Einschaltdauer der Null-

zustände durchgeführt werden. Durch die Prüfung der Bedingung der Fallunterscheidung wird festgestellt, ob die entsprechende Phase den Wert +1 oder -1 annehmen muß, d.h.

$$T_{Null} = \begin{cases} -\min(t_{0a}, t_{0b}, t_{0c}), & \min(t_{0a}, t_{0b}, t_{0c}) < T_{pH} - \max(t_{0a}, t_{0b}, t_{0c}) \\ T_{pH} - \max(t_{1a}, t_{1b}, t_{1c}), & \text{sonst.} \end{cases} \quad (4.26)$$

Die Umschaltzeitpunkte t_i berechnen sich dann wiederum nach Gleichung (4.25).

Aus c) in Bild 4.12 ist zu entnehmen, daß in diesem Fall das Ein- und Ausschalten einer Phase an den Halbperiodengrenzen zusammenfällt und sich so gegenseitig aufhebt. Der Zustand dieser Phase ändert sich also nicht, sodaß für dieses Beispiel (Aussteuerung der Phase a wird auf +1 gesetzt) der Nullspannungsraumzeiger 0U nicht mehr auftritt. Wird die Aussteuerung einer Phase auf den Wert -1 gesetzt, so tritt entsprechend der Nullspannungsraumzeiger 7U nicht mehr auf. Es ist leicht nachzuweisen, daß innerhalb eines 60° -Sektors, der jeweils symmetrisch um die Zustände 1 bis 6 liegt, immer dieselbe Phase unverändert bleibt, was dazu führt, daß nur noch zwei Phasen gepulst werden. Dadurch tritt eine Verringerung der Schaltverluste auf, da sich die Zahl der Umschaltungen von drei auf zwei, also um ein Drittel, reduziert. Mit einem als gegeben vorausgesetzten Stromrichter kann auf diese Weise die Schaltfrequenz um den Faktor 1,5 gesteigert werden, ohne daß höhere mittlere Schaltfrequenzen in den Bauelementen entstehen.

Auch hier sind bei umlaufenden Raumzeigern Modulationsgrade M bis zu $\sqrt{2/3}$ realisierbar, was einem Aussteuerungsgrad A von 0.907 entspricht.

Der relative Verlauf des Stromraumzeigers während der Pulsperiode, die c) in Bild 4.12 entspricht, ist in Bild 4.15 eingezeichnet. Die beiden Dreiecke, die zu den Halbperioden T_{pA} bzw. T_{pB} gehören, treffen nur noch an einer Spitze aufeinander.

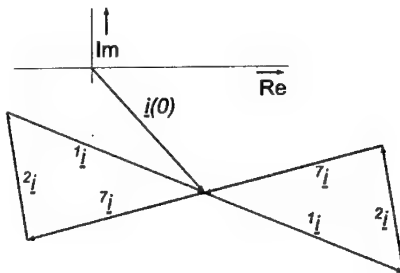


Bild 4.15: Relativer Verlaufs des Stromraumzeigers gemäß den Spannungsverläufen nach Bild 4.12 c)

Eine Variante dieses Verfahrens besteht nach [4.15] darin, den Bereich, in dem eine Phase nicht geschaltet wird, um maximal $\pm 30^\circ$ zu verschieben. So kann bei geeigneter Verschiebung vermieden werden, daß Umschaltungen im Bereich des Strommaximums auftreten, wodurch die Schaltverluste und die Oberschwingungen verringert werden. Es ist aber sinnvoll, diese Variante nur in analytischer Form zu verwenden.

4.3.6.4 Zweiphasiges Pulsen zweiter Art

In Anlehnung an den vorigen Abschnitt kann die Nullkomponente auch so gestaltet werden, daß die dem Betrag nach zweitgrößte der drei Phasenaussteuerungen, den Wert ± 1 annimmt. Für die Ermittlung des jeweils richtigen Vorzeichens ist hier wiederum eine entsprechende Fallunterscheidung bezüglich der Zeitauern der Nullzustände notwendig. Die zeitliche Verschiebung T_{Null} berechnet sich damit zu

$$T_{Null} = \begin{cases} T_{pH} - \max(t_{0a}, t_{0b}, t_{0c}), & \min(t_{0a}, t_{0b}, t_{0c}) > T_{pH} - \max(t_{0a}, t_{0b}, t_{0c}) \\ -\min(t_{1a}, t_{1b}, t_{1c}), & \text{sonst} \end{cases} \quad (4.27)$$

Wie auch im vorigen Abschnitt ergeben sich die neuen Umschaltzeitpunkte nach Gleichung (4.25). In d) von Bild 4.12 ist zu sehen, daß auch hier eine Phasenumschaltung entfällt. Im Gegensatz zum zweiphasigen Pulsen erster Art (Bild 4.12 c)) tritt für das dort gezeigte Beispiel (Aussteuerung der Phase c wird -1 gesetzt) der Nullspannungsraumzeiger \vec{U} nicht mehr auf. Ein weiterer Unterschied liegt in der Aufteilung derjenigen Bereiche, in denen die jeweilige Phase nicht schaltet. Während es beim zweiphasigen Pulsen erster Art zwei Bereiche zu je 60° sind, bilden sich hier in jeder Phase vier Bereiche zu je 30° aus. Diese Art der Modulation ist in der Literatur (z.B. [4.10]) auch unter der Bezeichnung "Modifizierte Zweiphasen-Modulation" beschrieben.

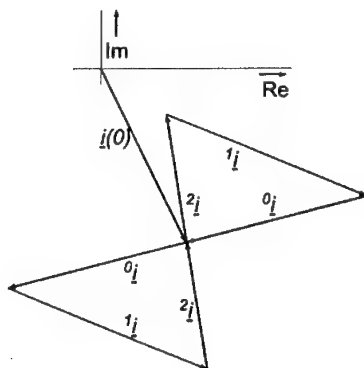


Bild 4.16: Verlauf des Stromraumzeigers nach den Spannungsverläufen in Bild 4.12 d)

Auch bei diesem Pulsverfahren treffen die beiden Dreiecke des relativen Stromraumzeigerverlaufs entsprechend Bild 4.16 nur noch an einer Ecke aufeinander. Es wurde in der Literatur (z.B. [4.10], [4.16]) bereits mehrfach beschrieben, daß dieses Verfahren über den gesamten Bereich der realisierbaren Aussteuerungsgrade geringere Oberschwingungen erzeugt als das zweiphasige Pulsen erster Art, während der Vorteil einer möglichen Pulsfrequenzerhöhung gegenüber der Einzelphasenmodulation weiterhin bestehen bleibt.

5 Stromrichtersteuerung

Bei der Anwendung des PC für die Steuerung und Regelung von Stromrichtern kann man drei wesentliche Aufgaben unterscheiden, auf welche die Rechenleistung aufgeteilt werden muß. Dies sind in der Reihenfolge der Wirkungskette die Aufgaben "Kommunikation", "Steuerung und Regelung" sowie "Schutz". Die zeitliche und funktionale Priorität dieser Funktionen ist jedoch in umgekehrter Reihenfolge zu sehen.

Die Kommunikation umfaßt alle Funktionen, die System- oder Betriebsdaten mit dem Bediener oder regulären Peripherieeinheiten (Drucker, etc.) austauschen. Denkbar sind hier beispielsweise Bildschirmausgaben (in Text und Graphik) oder das Einlesen und Verarbeiten von Zeichen aus der Tastatur. Ebenso können on-line erfaßte Meßdaten auch auf dem Drucker ausgegeben werden. Andererseits ist z.B. die Anbindung an Feldbussysteme oder lokale Netze möglich. Die meisten dieser Funktionen sind jedoch für den grundlegenden Betrieb von Stromrichtern nicht von Bedeutung. Die Realisierung einer einfachen Eingabemöglichkeit über die Tastatur wird in Abschnitt 6.1 beschrieben.

Die Aufgabe "Steuerung und Regelung" beinhaltet diejenigen Routinen, welche den Ablauf und die Berechnung der Phasenumschaltungen unter Einsatz von Hardwareinterrupts organisieren sowie diejenigen Routinen, die der Regelung des Antriebs dienen. In Stromrichtersteuerungen auf PC-Basis treten jedoch systembedingt besondere Betriebszustände auf, die in der Interruptsteuerung zusätzlich beachtet werden müssen. Deshalb wird in Abschnitt 5.2 zunächst der Fall behandelt, bei dem keine solchen Sonderfälle auftreten. Dann folgt die Beschreibung (Abschnitt 5.4) und Behandlung (Abschnitt 5.5) der möglichen Sonderfälle.

In analog aufgebauten Stromrichtersteuerungen werden die Umschaltzeitpunkte normalerweise durch den Vergleich einer Führungsspannung und einer Hilfsgröße (z.B. dreieck- oder sägezahnförmige Hilfsspannung bei Pulsverfahren oder auch sinusförmige Hilfsspannung bei netzgeführten Stromrichtern) ermittelt. Eine direkte Umsetzung dieser Methoden in die Digitaltechnik besteht darin, durch wiederholte Berechnungen in kurzen Zeitabständen zu ermitteln, ob die vorgegebene Schaltbedingung erfüllt ist oder nicht. Solche Verfahren werden durchaus realisiert, insbesondere um direkte Zweipunkt- oder Mehrpunktregler digital aufzubauen. Dabei werden aber sehr schnelle Rechner meist mit extrem häufiger Meßwerterfassung verlangt, so daß diese Methoden für die Steuerung mit dem PC praktisch ausscheiden.

Die zweite - inzwischen weitgehend etablierte - Möglichkeit besteht darin, die Umschaltzeitpunkte im voraus zu ermitteln. Die jeweils nächsten Zeitpunkte werden in Zeitwerke geladen, die sich bei Erreichen der eingestellten Zeit selbständig mittels eines Interrupts beim Rechner melden (Weckerprinzip) und dann die Umschaltung durch den Rechner veranlassen, der anschließend die Zeitwerke neu einstellt. Zwischen den Interrupts kann der Prozessor beliebige Routinen abarbeiten ohne sich um die Einhaltung der Zeitpunkte kümmern zu müssen.

Schließlich muß der Stromrichter, dessen Last und nicht zuletzt auch der Bediener vor möglichen Fehlfunktionen geschützt werden. Ein geeignetes Schutzkonzept wird in Abschnitt 7 vorgestellt. Dabei ist eine mehrstufige Ausführung mit Schnell- oder Direktschutz für den Schutz der Leistungshalbleiter bzw. weiteren, aber langsameren Schutzmechanismen möglich.

5.1 Interruptsteuerung

Im Stromrichterbetrieb werden Interrupts hauptsächlich dazu benutzt, um den Ablauf von Zeitintervallen zu erfassen. Dazu sind zunächst alle im PC integrierten Zeitwerke (z.B. der Systemtimer auf dem Motherboard, Zeitwerke auf Einsteckkarten usw.) geeignet. Allerdings ergeben sich für die Nutzung des Systemtimers wegen seiner festen Verschaltung auf dem Motherboard Einschränkungen bezüglich der Handhabung von Interrupts, so daß er für den Stromrichterbetrieb nicht in Betracht kommt. Da er aber im Stromrichterbetrieb mittels des ihm zugeordneten Hardwareinterrupts die Systemuhr immer wieder aktualisieren und somit unerwünschte Verzögerungen verursachen würde, wird die ihm zugeordnete Interruptleitung IRQ0 [5.1] über das Interrupt-Mask-Register (siehe Abschnitt 2.2.2) des Interrupt-Controllers maskiert und so der Systemtimer abgeschaltet. Somit werden im folgenden nur noch diejenigen Zeitwerke benutzt, die sich auf den Einsteckkarten befinden. Deren Programmierung und Handhabung unterscheidet sich jedoch nicht vom Systemtimer.

5.1.1 Multizähler-Verfahren

Bei diesem Verfahren wird jeder Phase ein eigenes Zeitwerk zugeordnet, das einen bestimmten Interrupt auslöst. Auf diese Weise müssen für alle drei Phasen sowohl Zeitwerke wie auch Interruptroutinen bereitgestellt werden. Allein aus dem Typ des Interrupts ist hier die Art (z.B. Phasenumschaltung) der anstehenden Aktion erkennbar. Durch eindeutige Verknüpfung von Zeitwerk und ausgeführter Routine wird die Erkennung oder Abfrage der anstehenden Umschaltung bereits vom System selbst ausgeführt und muß so nicht mehr programmiert werden. Allerdings können sich bei diesem Verfahren Prioritätsprobleme in Bezug auf die Bearbeitung

gleichzeitiger Steuereingriffe ergeben, was insbesondere auch deren Auffangen durch Sonder-routinen erschwert. Außerdem sind meist zusätzliche Einsteckkarten notwendig, da nur wenige Meßwerterfassungskarten drei oder mehr frei verfügbare Zeitwerke enthalten. Neben dem höheren Programmieraufwand für die Interruptroutinen selbst schränkt dieses Verfahren die möglichen zusätzlichen Funktionen des PC stark ein, da viele davon mit systemeigenen Interruptroutinen arbeiten, die jedoch durch die selbst programmierten Interruptroutinen ersetzt wurden.

Aus diesen Gründen ist dieses Verfahren für die Steuerung und Regelung von Stromrichtern mit PC wenig praktikabel und wurde nicht weiter verfolgt.

5.1.2 Monozähler-Verfahren

Hier kommt nur ein einziges Zeitwerk zum Einsatz. Das hat den Vorteil, daß alle Steuereingriffe (z.B. Halbperiodengrenzen oder Umschaltzeitpunkte) mit derselben Interruptleitung behandelt werden. Dadurch lassen sich ggf. sogar mehrere Stromrichter mit dem gleichen PC steuern. Spezielle Routinen zur Behandlung von Sonderfällen lassen sich in ein solches System ebenfalls leichter integrieren. Im Gegensatz zum Multizähler-Verfahren ist hier auch meist keine zusätzliche Einsteckkarte notwendig, da die meisten Multi-I/O-Karten über wenigstens ein frei verwendbares Zeitwerk verfügen. Außerdem können für die wichtigsten Peripheriegeräte die Originalfunktionen des Betriebssystems verwendet werden¹, da nur eine einzige Interruptleitung durch das Zeitwerk belegt ist. Dieses Verfahren wird deshalb in dieser Arbeit angewandt.

Zunächst wird aber angenommen, daß keine der oben erwähnten Sonderfälle auftreten. Diese werden in den Abschnitten 5.4 und 5.5 behandelt.

Die Information über die umzuschaltende Phase und ggf. auch die Richtung der Umschaltung muß beim Monozähler-Verfahren softwaretechnisch in Form von Variablen und Abfrageroutinen abgespeichert werden. Wie in Abschnitt 4.2 eingeführt, erfolgt die Notation der Zustände eines Phasenschalters üblicherweise mit Hilfe einer einstelligen Binärzahl. Hier ist es jedoch sinnvoll, beide Transistoren eines Phasenschalters in die Notation einzubeziehen. Dadurch erweitert sich die Zustandszahl für jede Phase auf zwei Stellen. Die Zustandsbeschreibung des dreiphasigen Wechselrichters umfaßt also 6 Stellen (c+, c-, b+, b-, a+, a-), wovon die Bits 0 und 1 die Transistoren T4 und T1 der Phase a (siehe Bild 4.5), die Bits 2 und

¹Einige Interrupts müssen allerdings im Stromrichterbetrieb unterdrückt werden (z.B. Systemtimer)

3 die Transistoren T6 und T3 der Phase b sowie die Bits 4 und 5 die Transistoren T2 und T5 der Phase c angeben. Der neue Stromrichterzustand nach einer Umschaltung wird nun mit Hilfe der logischen Funktion "XOR" (bitweises exklusiv-ODER) ausgerechnet, wobei ein Operand den aktuellen Stromrichterzustand und der andere Operand die gewünschte Zustandsänderung darstellt. Tabelle 5.1 zeigt die Wahrheitstafel dieser Funktion. Im Normalbetrieb sind die Werte der Stellen 0,1 sowie 2,3 und 4,5 jeweils invers zueinander¹. Besitzt z.B. der momentane Zustand des Stromrichters den Wert "10 01 10" und es soll die Phase "c" umgeschaltet werden, so ist die zugehörige Zustandsänderung "11 00 00" und der neue Stromrichterzustand ergibt sich zu

$$(10\ 01\ 10) \text{ XOR } (11\ 00\ 00) = 01\ 01\ 10.$$

Diese Art der Darstellung ermöglicht es auch Doppel- oder Dreifachumschaltungen mit derselben Routine und demselben Zeitaufwand zu bearbeiten. Dies wird aber nur in Sonderfällen (z.B. "Vereinigungsmethode" Abschnitt 5.5.2.1) benutzt. Weitere Sonderfälle treten beispielsweise beim Ingangsetzen und Abschalten des Stromrichters auf, da der Ruhezustand durch 00 00 00 gekennzeichnet ist. Dies ist im Abschnitt 5.5.3 beschrieben.

Tabelle 5.1:
Wahrheitstabelle der "Exklusiv-ODER"-Funktion

	$x_1 = 0$	$x_1 = 1$
$x_2 = 0$	0	1
$x_2 = 1$	1	0

5.2 Steuereingriffe durch Interrupts

5.2.1 Berechnungsalgorithmus

In Abschnitt 4.1 wurde bereits erwähnt, daß für die Benutzung von prädiktiven Pulsverfahren - wie sie hier angewandt werden - alle für einen zukünftigen Zeitraum relevanten Steuereingriffe bereits vor dessen Beginn bekannt sein müssen. Im Prinzip ist es gleichgültig, ob sich ein solcher Zeitraum über eine Halbperiode, eine Vollperiode oder Vielfache davon hinzieht. Auch die Dauer der einzel-

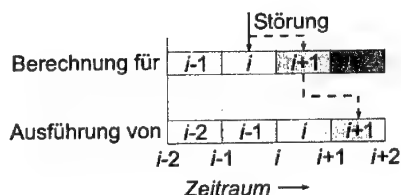


Bild 5.1: Darstellung der Berechnungs- und der Ausführungszeiträume der Beschreibungstabelle

¹Die notwendigen Sicherheitszeiten beim Umschalten mit den Werten "00" in einer Phase werden entsprechend Abschnitt 3.3 hier von den Ansteuermodulen eingefügt. Sie lassen sich jedoch auch durch eine Software routine einbauen.

nen Zeiträume kann variieren. Dies führt zu einer Aufteilung zwischen dem Berechnungszeitraum und dem Ausführungszeitraum, wie er in Bild 5.1 dargestellt ist. Im jeweils vorherigen Zeitraum werden die notwendigen Daten für den nachfolgenden Zeitraum berechnet. Das hat aber zur Folge, daß Reaktionen auf äußere Steuereingriffe (Störgrößen und Stellgrößen) frühestens im folgenden Zeitraum in die Berechnung eingehen und im übernächsten ausgeführt werden können; denn Veränderungen des aktuellen, gerade auszuführenden Datensatz sind nur bedingt möglich¹. Durch Überschneidungen hinsichtlich der Reihenfolge der Steuereingriffe - insbesondere neuer Phasenumschaltungen - könnten sich unzulässige Betriebszustände (siehe Abschnitt 5.4 und 5.5) ergeben. Hier werden Berechnungszeitraum und Ausführungszeitraum mit der jeweiligen Halbperiode gleich genutzt.

Bild 5.2 zeigt die prinzipielle Vorgehensweise der Pulsumsetzung. Unter a) ist der Verlauf der Zustände z_a, z_b, z_c (Bild 5.2 a)) der Transistoren T_1, T_3, T_5 aus Bild 4.5 eingezeichnet. Die Zeitpunkte der Zustandsänderungen stellen drei von einer gesamten Anzahl j_{max} im zu berechnenden Zeitraum auftretenden Steuereingriffen dar. Sie sollen in den Abständen $T_a, T_b, T_c, \dots, T_{j_{max}}$ vom Beginn des Ausführungszeitraums ($t = 0$)² auftreten (Bild 5.2 b)) und müssen in der richtigen Reihenfolge verarbeitet werden (Bild 5.2 c)). Alle Steuereingriffe werden

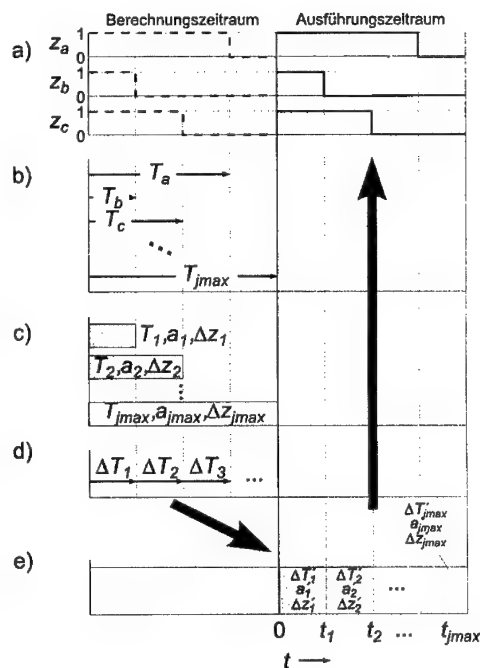


Bild 5.2: Umsetzung einer Abfolge von Steuereingriffen im Einzel-Timer-Verfahren

¹Die Verzögerung auch damit verknüpft, daß möglichst hohe Pulsfrequenzen realisiert werden sollen, bei denen die Berechnung das Umschaltzeitpunkte einen deutlichen Teil des betrachteten Zeitraums in Anspruch nimmt. Bei niedrigen Pulsfrequenzen und auch bei netzgeführten Stromrichtern kann davon abgewichen werden.

²Da jeder Zeitraum unabhängig vom vorhergehenden betrachtet wird, wird die Zeit-zählung mit jedem neuen Zeitraum wieder bei 0 begonnen.

mindestens durch den Abstand zum Anfang des Zeitraums, die Art des Steuereingriffs¹ und die Zustandsänderung des Stromrichters, die der Eingriff hervorruft definiert. Ggf. können auch weitere Daten solche Steuereingriffe beschreiben. Zunächst bleibt die Definition aber auf die drei genannten beschränkt. Dadurch ergibt sich eine dreielementige Datenstruktur. Zur besseren Handhabung werden alle j_{max} Datenstrukturen in einer Tabelle zusammengefaßt (siehe Tabelle 5.2). Die drei Elemente einer Tabellenzeile sind also:

1. Steuereingriff-Zeitpunkt T
2. Art des Steuereingriffs a
3. Steuereingriff-Zustandsänderung Δz ²

Die Variable j bezeichnet dabei den Laufindex des jeweiligen Steuereingriffs und ist nicht explizit in der Tabelle gespeichert.

Tabelle 5.2:
Struktur der beschreibenden On-line-Tabelle eines Zeitraums

j	T	a	Δz
1	T_1	a_1	Δz_1
2	T_2	a_2	Δz_2
\vdots	\vdots	\vdots	\vdots
j_{max}	t_{jmax}	a_{jmax}	Δz_{jmax}

Die zeitliche Reihenfolge der Vorgänge ist zu Beginn der Prädiktion unbekannt. Deswegen werden außer dem Steuereingriff, der das Ende des Berechnungszeitraums darstellt, alle anderen in willkürlich festgelegter Reihenfolge berechnet und in die Tabelle eingetragen. Der Steuereingriff für das Ende des Berechnungszeitraums wird immer als letzter auftretender Eingriff mit der Nummer j_{max} in der Tabelle abgespeichert, selbst wenn ein Steuereingriff erst nach dem Halbperiodenende auftreten würde (vgl. Abschnitt 5.5.3). Die Tabelle wird dann in einem zusätzlichen Schritt zeitlich geordnet.

Wegen der Aufteilung in Berechnungs- und Ausführungszeitraum werden zwei verschiedene On-line-Tabellen (Berechnungstabelle, Ausführungstabelle) für beide Zeiträume angelegt, wobei am Ende eines Ausführungszeitraums die Ausführungstabelle mit der Berechnungstabelle überschrieben wird³.

¹Mögliche Arten von Steuereingriffen können Umschaltungen, die Grenzen der Ausführungszeiträume oder Eingriffe für Messungen darstellen. So können selektiv zusätzliche Unterprogramme in der Interruptroutine aufgerufen werden.

²In der tatsächlich verwendeten Tabelle wird die Zustandsänderung in Übereinstimmung mit Abschnitt 5.1.2 als sechsstellige Binärzahl für alle sechs Transistoren angegeben.

³Im Programm erfolgt das "Überschreiben" durch Austauschen der Tabellen mittels Zeiger

Für die Anwendung des Monozähler-Verfahrens werden die Zeitdauern T_1, T_2, T_3 , usw., die vom Beginn des Ausführungszeitraums gezählt werden, umgewandelt in Intervalle ΔT_j , die beim letzten vorhergehenden Steuereingriff beginnen (Bild 5.2 d)).

$$\Delta T_j = T_j - T_{j-1} \quad , \quad j = 1, 2, \dots, j_{max} \quad (5.1)$$

Der Zeitpunkt T_0 bezeichnet dabei den Beginn des Berechnungszeitraums, d.h. $T_0 = 0$. Die neuen Werte werden noch vor Beginn des Ausführungszeitraums in die Tabelle geschrieben, die dann eine Form nach Tabelle 5.3 annimmt.

Tabelle 5.3:
Struktur der Ausführungstabelle

j	T	a	Δz
1	ΔT_1	a_1	Δz_1
2	ΔT_2	a_2	Δz_2
\vdots	\vdots	\vdots	\vdots
j_{max}	t_{jmax}	a_{jmax}	Δz_{jmax}

5.2.2 Interruptroutine

Die Steuereingriff-Intervalle ΔT_j werden im Laufe des Ausführungszeitraums nun nacheinander in das Zeitwerk geladen. Nach jedem Ablauf eines solchen Intervalls löst das Zeitwerk einen Interrupt aus. In der zugehörigen Interruptroutine (siehe Bild 5.3) wird zunächst der Steuereingriff gemäß der Laufvariablen j ausgeführt. Dadurch wird die Zeitverzögerung vom Beginn des Interrupts bis zum tatsächlichen Steuereingriff klein gehalten. Dann wird der Steuereingriff für das nächste Intervall $j+1$ vorbereitet. Dies beinhaltet das Laden des Zeitwerks für das nächste Intervall sowie die Erhöhung des laufenden Index auf $j+1$. Anschließend können noch je nach Art des Interrupts (siehe Beschreibungstabelle) spezifische Zusatzroutinen folgen, wie beispielsweise die Erneuerung der On-line-Tabelle am Ende des Ausführungszeitraums. Schließlich muß die Interruptroutine für die Rücksetzung des "End-of-Interrupt"-Flags des Interrupt-Controllers sorgen.

Insgesamt sollten aber nicht zu viele Aufgaben an die Interruptroutine delegiert werden. Da diese ständig von Zeitwerk aufgerufen wird, können sich dadurch unnötige Verzögerungen im Prozeßablauf ergeben (siehe Abschnitt "Sonderfälle").

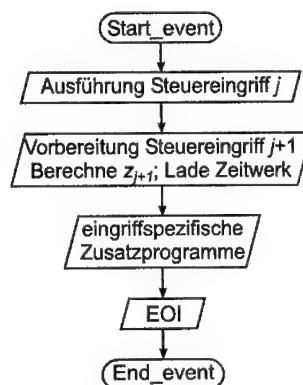


Bild 5.3:
Programflußplan der Interrupt-
routine
EOI="End-of-Interrupt"

5.3 Spezifische Zusatzroutinen

5.3.1 Phasenumschaltung

Die Ausgabe des Stromrichterzustandes findet bei jedem Steuereingriff statt. Diese Vorgehensweise ist deshalb notwendig, weil durch die Behandlung von Sonderfällen (siehe Abschnitt 5.5) auch Steuereingriffe auftreten können, die ursprünglich zwar nicht der Phasenumschaltung dienen, die eine solche Änderung aber dennoch herbeiführen müssen (z.B. Ende des Ausführungszeitraums und Umschaltung). Damit ist aber auch die wichtigste Aufgabe von Steuereingriffen dieser Art schon erfüllt. Eine weitere Behandlung ist nicht notwendig.

5.3.2 Ende eines Ausführungszeitraums

Bild 5.4 zeigt den Programmflußplan der Zusatzroutine für diesen Fall. Nach Ablauf des Intervalls ΔT_{jmax} wird die Ausführungstabelle mit neuen Werten belegt sowie der interne Intervallzähler für den Index auf den Wert 0 zurückgesetzt. Nach dem Umladen der Berechnungstabelle in die Ausführungstabelle kann nun die Berechnungstabelle ihrerseits wieder mit Daten beschrieben werden. Deshalb wird ein Berechnungsflag gesetzt, das nach dem Rücksprung in das Hauptprogramm wieder neue Daten anfordert.

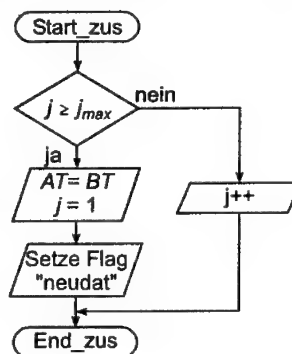


Bild 5.4:
Programmflußplan eines
Steuereingriffs vom Typ "Ende
des Ausführungszeitraums"

5.3.3 Meßdatenerfassung

Die Erfassung von analogen Signalen über die Multi-I/O-Karte ist vergleichsweise einfach. Mit der Konfiguration nach Abschnitt 3 ist lediglich der Befehl zum Start der Messung notwendig. Das wird mit einem Schreibbefehl in das entsprechende Register der Multi-I/O-Karte realisiert. Nach Abschnitt 4 ist das Ende des Ausführungszeitraums der beste Zeitpunkt für eine Messung.

5.4 Sonderfälle

Sonderfälle treten immer dann auf, wenn die zeitpunktgenaue Ausführung der Interruptroutinen trotz korrekter Berechnung nach Abschnitt 5.2 aufgrund von PC-spezifischen Systemeigenheiten

genschaften nicht mehr gewährleistet ist. Dies tritt insbesondere bei Abweichungen der Systemzeit des Steuergeräts von der tatsächlichen Zeit zutage, sodaß zusätzliche Maßnahmen ergriffen werden müssen.

5.4.1 Verzögerungen durch die Laufzeit von "outp"-Befehlen

Mit Hilfe des "outp"-Befehls der Programmiersprache C¹ können Daten in den I/O-Adreßraum (siehe Abschnitt 2.1 und 2.2.1) geschrieben werden. Sie sind also z.B. zur Ausgabe der Stromrichterzustände und damit für den Stromrichterbetrieb zwingend notwendig². Anhand des Schaubilds des AT-Modells (Bild 2.1) wird aber deutlich, daß an der Ausgabe mehrere Komponenten beteiligt sind. In

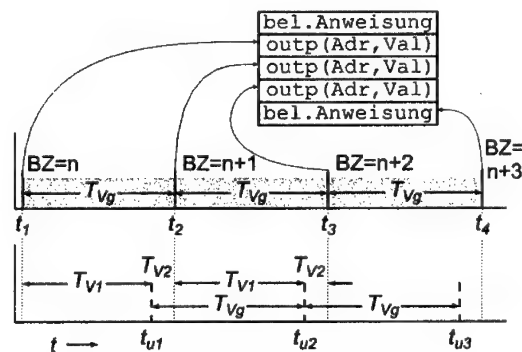


Bild 5.5: Zeitlicher Ablauf eines outp-Befehls am Beispiel der Umschaltung eines I/O-Ports der Multi-I/O-Karte (BZ: Befehlszeiger)

Bild 5.5 ist schematisch der zeitliche Ablauf der Bearbeitung von outp-Befehlen am Beispiel der Ausgabe eines Umschaltbefehls an den Stromrichter dargestellt. Zunächst muß der Prozessor die Zieladresse "Adr" richtig interpretieren und der PCI- oder ISA-Brücke zuführen. Auch die Multi-I/O-Karte muß die ankommende Adresse erst dekodieren, bevor der Wert ("Val") an den entsprechenden Baustein weitergeleitet wird, der es schließlich an den Stromrichter ausgibt. Danach müssen die benutzten Busleitungen wieder freigegeben und der Befehlszeiger BZ um 1 erhöht werden. Es entsteht also zunächst eine Verzögerung T_{V1} zwischen dem Laden des outp-Befehls in die CPU (t_1, t_2, t_3) und dem tatsächlichen Schreiben des Werts in die Adresse (t_{u1}, t_{u2}, t_{u3}). Die anschließende Freigabe der Busse verursacht eine weitere Verzögerung T_{V2} . Die Anteile der einzelnen Komponenten (z.B. PCI-Brücke) an den Verzögerungen kann aufgrund der hohen Integrationsdichte der Bauteile auf dem Motherboard meßtechnisch nicht ohne weiteres erfaßt werden. Im übrigen sind auch weniger die einzelnen

¹In anderen Hochsprachen existieren ähnliche Befehle, z.B. Port [Adr] := Val in Pascal

²Es existieren auch Befehle zum Einlesen von Daten aus Basisadressen. Im weiteren werden Anweisungen, die dem Ein- oder Auslesen von Basisadressen dienen, allgemein als Ein-/Ausgabebefehle bezeichnet.

Anteile als vielmehr deren Summe von Interesse. Sie wird durch die Zeitdauer T_{V_z} beschrieben und als konstant angenommen.

Mit Hilfe eines Testprogramms, das auf dem Prinzip der wiederholten Ausführung basiert, konnte die Laufzeit von outp-Befehlen zu ca. $0,6 \mu s$ ermittelt werden. Dieser Wert bestätigte sich bei Messungen einzelner Vorgänge mit dem Oszilloskop. Da in Interruptroutinen nach Abschnitt 5.2.2 mindestens vier outp-Befehle (2x Zeitwerk, 1x Zustandsausgabe, 1x EOI) auftreten, führt dies zu einer deutlichen Verlängerung der Laufzeit der Interruptroutinen. Eine Verkürzung der Interruptroutinen ist aber nicht möglich. Allerdings kann wie in Abschnitt 5.4.2 beschrieben die Laufzeit der Interruptroutine als solches kompensiert werden, was die Berücksichtigung der outp-Befehle automatisch beinhaltet.

5.4.2 Verzögerungen durch Interrupts

Für viele Rechner-Anwendungen ist die Annahme zulässig, daß Interrupts keine Verzögerungen im Programmablauf verursachen. Dies gilt z.B. dann, wenn der Benutzer die geschwindigkeitsbegrenzende Komponente ist, wie beispielsweise in der Textverarbeitung. Bei Betrieb von Stromrichtern sind die Verzögerungen jedoch nahezu in der Größenordnung der Prozeßzyklen (z.B. Halbperiodendauern). Folglich können sie dann nicht mehr als beliebig kurze Unterbrechungen des Hauptprogramms angesehen werden, die keinen Einfluß auf die Programmlaufzeit haben. Deshalb muß die Bearbeitungszeit von Interruptroutinen selbst berücksichtigt werden. Das gilt insbesondere bei der Steuerung und Regelung von Stromrichtern mit PC, da hier die Bearbeitung von Interrupts wegen des allgemeinen, offenen Konzepts deutlich länger dauert als bei spezialisierten Mikrocontrollern¹.

Die direkte Bestimmung der Verzögerungszeiten ist nicht ohne weiteres möglich. Die Anschlüsse von Prozessor und Interruptcontroller auf dem Motherboard sind so schwer zugänglich, daß direkte Messungen nur mit aufwendigen Umbauten möglich sind. Deswegen wurden indirekte Verfahren benutzt.

Bild 5.6 zeigt den Verlauf zweier gut zugänglicher Signale. Sie werden durch ein Testprogramm hervorgerufen, das von einer Interruptroutine entsprechend Bild 5.3 unterbrochen wird und vom Typ "Phasenumschaltung" ist. Zum einen wurde im oberen Bild die Spannung u_T am Ausgang des Zeitwerks oszilloskopiert. Sie ist direkt am Baustein selbst auf der Multi-

¹Die Laufzeit von Interruptroutinen hängt aber auch vom Binärkode ab, den der Compiler erzeugt. Um Aussagen darüber treffen zu können, müßten verschiedene Compiler daraufhin untersucht werden. Das ist aber nicht Gegenstand dieser Arbeit.

I/O-Karte abgreifbar. Beim Ablauf des eingestellten Intervalls springt diese Spannung von 0V auf 5V und löst so den Interrupt aus. Erst mit dem erneuten Einstellen des Zeitwerks durch die Interruptroutine springt das Signal zurück auf 0V. Im unteren Bild wurde durch geringfügige Modifikation der Interruptroutine deren Laufzeit sichtbar gemacht. Dazu wurde vor deren erster Anweisung ein Befehl zur Umschaltung eines freien I/O-Ports, d.h. u_{out} springt von 0V auf 5V, eingefügt. Nach der letzten Anweisung der Routine ("EOI") wurde mit Hilfe eines weiteren Umschaltbefehls derselbe Port wieder auf 0V zurückgestellt.

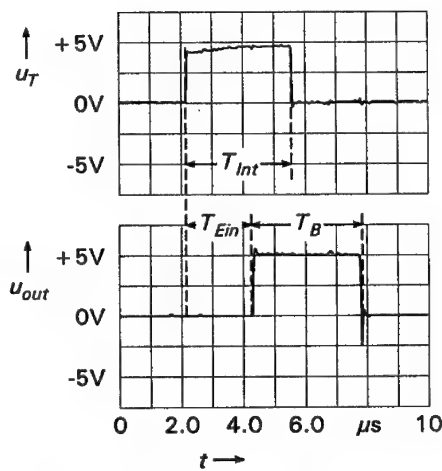


Bild 5.6: Bearbeitungszeit, Einsprungszeit und Interruptzeit eines PentiumII-AT-Rechners mit 400 MHz

Gut zu erkennen sind drei Zeitdauern, die im folgenden von Interesse sind. Das ist die "Interruptzeit" T_{Int} , die "Einsprungszeit" T_{Ein} sowie die "Bearbeitungszeit" T_B . Die Interruptzeit T_{Int} bezeichnet dabei die Zeitdauer zwischen dem Ende eines Intervalls und dem erneuten Einstellen des Zeitwerks. Die Einsprungszeit T_{Ein} definiert die Zeitdauer von Ablauf eines Intervalls bis zum Beginn der Codeverarbeitung der Interruptroutine. Die Bearbeitungszeit T_B schließlich zeigt die Laufzeit des Codes der Interruptroutine an. Nicht zu sehen ist die "Ausprungszeit" T_{Aus} , die vom Ende der Bearbeitung der Interruptroutine bis zur Wiederaufnahme der unterbrochenen Routine reicht.

Interruptzeit T_{Int} : Sie bewirkt eine Verschiebung zwischen dem Steuerungszeitsystem und dem Zeitsystem des Stromrichters, da das Steuerungszeitsystem während dieser Zeitspanne "angehalten" ist, das Zeitsystem des Stromrichters jedoch weiterläuft (es fließt ja auch weiterhin Strom!) und ist während des Steuerungsablaufs nicht erfassbar. Die Dauer eines Steuerungseingriff-Intervalls verlängert sich dadurch auf

$$\Delta T'_j = \Delta T_j + T_{Int} \quad , \quad (5.2)$$

und die Länge des Ausführungszeitraums auf

$$T'_{Zeltraum} = T_{Zeltraum} + j_{max} T_{Int} \quad . \quad (5.3)$$

Dies wird im folgenden "Verzögerung erster Art" (siehe Abschnitt 5.5) genannt.

Einsprunzeit T_{Ein} : Diese Verzögerung des Bearbeitungsbeginns der Interruptroutine wird durch weitgehend systeminterne Vorgänge hervorgerufen. Da der Interruptcontroller bei modernen AT-Rechnern selbst nicht mehr zugänglich ist (vgl. oben), wird das Ausgangssignal u_7 des Zeitwerks über den entsprechenden Anschlußpunkt IRQx des ISA-Slots in das System zurückgeführt. Dort wird es über die ISA- und PCI-Brücke an den Interruptcontroller weitergeleitet. Dieser initiiert die Interruptsequenz wie in Abschnitt 2.3.2 beschrieben. Damit sind - ähnlich wie beim Ein-/Ausgabebefehlen - zahlreiche Hardwarekomponenten an der Erkennung und der Ausführung eines Interrupts beteiligt. Die Einsprunzeit bedeutet eine Verschiebung der Zeitsysteme zwischen dem Zeitwerk einerseits und der Stromrichtersteuerung andererseits. Da die Zeitverschiebung weitgehend konstant ist und sowohl bei der Ausgabe der Umschaltungen wie auch beim Start der Messungen auftritt, ist sie für die Regelung nicht weiter schädlich und wird deshalb nicht weiter behandelt. Sie bleibt in den Steuerungen dieser Arbeit unberücksichtigt.

Bearbeitungszeit T_B : Diese Größe ist wesentlich vom behandelten Prozeß abhängig, da in den Interruptroutinen eine unterschiedliche Anzahl von Anweisungen bearbeitet werden müssen, um ihrer Aufgabe gerecht zu werden. Das gilt insbesondere für die eingriffsspezifischen Zusatzroutinen. Sie werden bei regulären Interrupts (siehe Bild 5.3) jedoch erst nach dem Laden des Zeitwerks aufgerufen, so daß deren Laufzeit dort ohne Auswirkung bleibt. Die in Abschnitt 5.5 beschriebenen "Maßnahmen zweiter Art" erfordern jedoch eine genauere Behandlung der Laufzeit.

Aussprunzeit T_{Aus} : Während dieser Zeit stellt der Prozessor wieder seine Register her, wie sie vor der Unterbrechung durch den Interrupt gültig waren. Daran sind nur der Prozessor und der Arbeitsspeicher beteiligt, sodaß die Aussprunzeit kürzer sein dürfte als die Einsprunzeit. Sie wird durch eine pauschale Vergrößerung von T_{min} um ca. 2 μs in der Steuerung berücksichtigt (siehe Abschnitt 5.5.2).

5.5 Maßnahmen für Sonderfälle

5.5.1 Maßnahmen erster Art

Nach Gleichung (5.2) wirken die Werte ΔT_j in der Berechnungstabelle in der Realität als $\Delta T_j'$. Zur Kompensation werden sie um die Dauer T_{int} reduziert. Dadurch können auch negative

Intervalle ΔT_j entstehen; sie werden durch die in Abschnitt 5.5.2 beschriebenen Sonder Routinen bearbeitet.

Der Betrag der Interruptzeit T_{int} ist während des Ablaufs der Steuerung nicht meßbar. Er ändert sich während des Stromrichterbetriebs nicht, hängt aber stark vom verwendeten PC-System ab. Trotzdem soll die Steuerung für verschiedene Konfigurationen der Hardware geeignet sein. Deswegen ist beim Start des Systems eine "Kalibrierungsroutine" zur Messung der verschiedenen benutzten Systemzeiten vorgesehen.

Zur Ermittlung der Interruptzeit T_{int} wird ein zweifach Interruptsystem entsprechend Bild 5.7 benutzt. Der erste Timer wird auf ein Vielfaches der erwarteten Interruptzeit T_{int} eingestellt. Die zugehörige Interruptroutine enthält als einzige Anweisung nur das Setzen des "Ende"-Merkers, der im Hauptprogramm in einer Warteschleife abgefragt wird.

Der zweite Interrupt wird

vom zweiten Zeitwerk gesteuert und ruft die im späteren Betrieb vorgesehene Interruptroutine ("Interrupt 2") auf, die wiederholt ausgelöst wird. Im Gegensatz zum späteren Einsatz wird hier aber immer wieder dasselbe Intervall in das Zeitwerk geladen. Es muß klein genug sein, um innerhalb der Meßdauer oft genug ausgelöst werden zu können und damit ggf. auftretende Meßfehler gering zu halten. Andererseits muß es groß genug sein, so daß der Interrupt in dieser Zeit sicher ausgeführt werden kann. Im vorliegenden Fall wurde das Interruptsystem mit zwei Zeitwerken vom Typ 8254 aufgebaut. Bei einer Taktung des Zeitwerks von 10 MHz kann jedes eine Zeitdauer von maximal 6,5335 ms mit einer Auflösung von 100 ns darstellen. Bei einem festen Steuereingriff-Intervall von $T_{event} = 10 \mu s$ ergibt sich damit maximale Anzahl an ausgelösten Interrupts von $n_{max} = 653$ Interrupts, wenn die Interruptzeit $T_{int} = 0$ wäre. Aus der Anzahl n_{ist} der tatsächlich ausgelösten Interrupts läßt sich dann die Interruptzeit T_{int} ausrechnen, um welche die Steuereingriff-Intervalle gekürzt werden müssen.

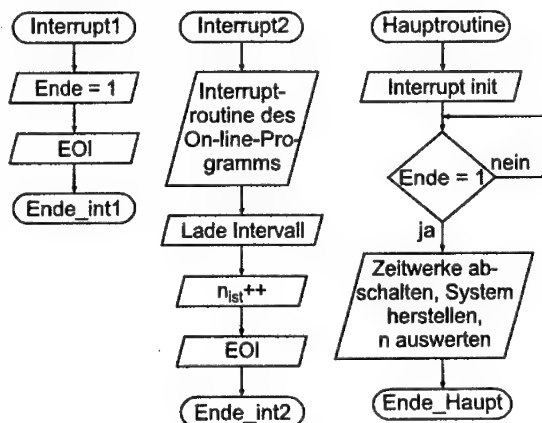


Bild 5.7: Programmlaufpläne eines zweifachen Interruptsystems zur Ermittlung der Interruptzeit T_{int}

$$T_{Int} = \left(\frac{n_{max} T_{event}}{n_{ist}} \right) - T_{event} \quad (5.4)$$

Mit diesem Verfahren werden schädliche Systemeinflüsse (z.B. zusätzliche Verzögerungen durch nicht erfaßbare Interrupts der Systemuhr) vermieden. Es müssen aber dafür annähernd ähnliche Verhältnisse geschaffen werden, wie sie auch im tatsächlichen Betrieb auftreten. Deswegen werden die Interrupts, die auch später nicht auftreten, hier ebenfalls unterdrückt. Die ermittelte Zeit wird in einer Variablen (Typ "float") gespeichert und im Betrieb zur Kompensation benutzt.

5.5.2 Maßnahmen zweiter Art

Im folgenden wird angenommen, daß die Einträge der On-line-Tabellen bereits hinsichtlich Verzögerungen erster Art kompensiert wurden. Bedingt durch die realen Interruptzeiten kann eine zweite Erscheinungsform von Verzögerungen auftreten. Denn zwei aufeinander folgende Interrupts können nur dann zeitpunktgenau bearbeitet werden, wenn sie weiter als die Mindestzeit T_{min} auseinander liegen. Diese setzt sich zunächst zusammen aus

- T_{Int} , d.h. den maßgeblichen Anteilen der Einsprungrzeit T_{Ein} und der Bearbeitungszeit T_B sowie
- der Aussprungrzeit T_{Aus} .

Die Aussprungrzeit wird dabei durch einen pauschalen Zuschlag von 2 μs berücksichtigt. Bild 5.8 a) zeigt eine mögliche Folge von Steuereingriff-Zeitpunkten (t_1, t_3, t_4), wie sie bei Stromrichterbetrieb auftreten. Die Steuereingriffe zu den Zeitpunkten t_1 und t_3 liegen weiter als die Mindestzeit T_{min} auseinander, d.h. $\Delta T_1 \geq T_{min}$. Die Interruptroutine zum Zeitpunkt t_1 kann also regulär bearbeitet

werden. Im Gegensatz dazu liegen die Zeitpunkte t_3 und t_4 näher beieinander ($\Delta T_2 < T_{min}$), so daß die zum Zeitpunkt t_4 gehörige Interruptroutine zu spät käme. Es werden zwar beide Steuereingriffe registriert, allerdings können sie nur seriell bearbeitet werden. Die Ausführung eines zweiten Interrupts kann also frühestens zum Zeitpunkt t_5 stattfinden (siehe Bild 5.8 b)). Dadurch ergäbe sich eine Fehlerzeit T_F . Sie ist immer auf den einzelnen Steuereingriff bezogen. Mit steigender Anzahl dicht aufeinander folgender Interrupts, summieren sich auch die Fehlerzeiten. Aufgrund der geschätzten Gesamtdauer eines Interrupts von ca. 6 μs ergeben sich dadurch ggf. Verzögerungen, die für den Betrieb nicht mehr akzeptabel sind [5.2]. Im folgenden werden zwei Wege zur Verbesserung vorgestellt.

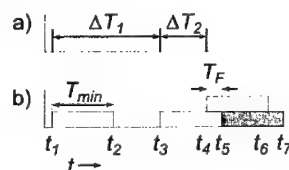


Bild 5.8:
Beispiel einer Verzögerung zweiter Art

5.5.2.1 Vereinigungsmethode

Bei dieser Methode werden Folgen von Steuereingriffen - wie sie in Bild 5.8 (t_3 bis t_8) gezeigt werden - durch einen Steuereingriff gleichzeitig ausgeführt. Die Entscheidung, welche Steuereingriffe wie vereinigt werden sollen, sowie die Vereinigung selbst werden bereits zur Berechnungszeit abgearbeitet.

Nach dem Aufstellen der Berechnungstabelle nach Abschnitt 5.2 ergäbe sich beispielsweise eine Verteilung der Steuereingriffe, wie sie in Bild 5.9 a) gezeichnet ist. Im Anschluß an die Berechnung wird Steuereingriff-Intervall für Steuereingriff-Intervall geprüft, ob die Bedingung für eine Verzögerung zweiter Art, d.h.

$$\Delta T_j < T_{min}, \quad j = 1, 2, \dots, j_{max} \quad (5.5)$$

erfüllt ist. Für diesen Fall wird nach Bild 5.9 b) der Steuereingriff $j-1$ mit dem Steuereingriff j vereinigt. Dazu wird das Steuereingriff-Intervall ΔT_j zum vorherigen Intervall ΔT_{j-1} hinzuaddiert, d.h.

$$\Delta T_{j-1}' = \Delta T_{j-1} + \Delta T_j, \quad j = 1, 2, \dots, j_{max} \quad (5.6)$$

Dadurch wird der Steuereingriff $j-1$ gleichzeitig mit dem Eingriff j ausgeführt. Die so neu entstandene Steuereingriff-Zustandsänderung $\Delta z_{j-1}'$ errechnet sich durch die Anwendung der XOR-Funktion auf die Operanden Δz_j und Δz_{j-1} .

$$\Delta z_{j-1}' = \Delta z_j \text{ XOR } \Delta z_{j-1}, \quad j = 1, 2, \dots, j_{max} \quad (5.7)$$

Die Werte Δz_{j-1} und ΔT_{j-1} der Zeile $j-1$ der Berechnungstabelle werden durch die Werte $\Delta z_{j-1}'$ und $\Delta T_{j-1}'$ ersetzt. Anschließend werden alle nachfolgenden Zeilen $j+1, j+2, \dots, j_{max}$ der Berechnungstabelle um eine Position nach vorn gerückt. Dann erst erfolgt die Prüfung des nächsten Intervalls nach Gleichung (5.5) mit der eben beschriebenen Vorgehensweise.

Mit jedem Vereinigungsvorgang verringert sich die maximale Anzahl an Steuereingriffen, d.h. Interrupts, um 1. Zur Zählung der durchgeführten Vereinigungen wurde die Interrupt-Reduktionsvariable sw_redN eingeführt, die mit jeder Vereinigung um 1 erhöht wird. Mit Hilfe von sw_redN kann das Entscheidungskriterium " $j \geq j_{max}$ " nach Bild 5.4 für das Erreichen des Endes eines Zeitraums für diese Methode modifiziert werden. Ansonsten würde der Arbeitsdatensatz zum falschen Zeitpunkt ausgetauscht, was unzulässige Betriebszustände zur Folge haben könnte. Die neue Bedingung für das Endes des Zeitraums lautet jetzt

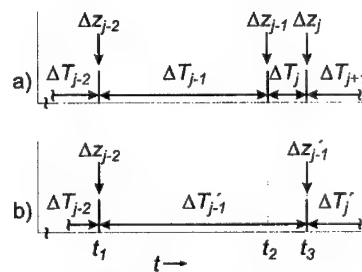


Bild 5.9: Vereinigungsmethode

$$j \geq j_{max} - sw_redN \quad (5.8)$$

Eine spezielle Behandlung erfordert der Fall, daß das erste Steuereingriff-Intervall ΔT_1 die Gleichung (5.5) erfüllt (siehe Bild 5.10 a)), da dies auch das Ende eines Ausführungszeitraums betrifft. Werden dafür die Gleichungen (5.6) und (5.7) angewandt, so würde dadurch das Ende des laufenden Zeitraums, das nach Abschnitt 4 auch der günstigste Zeitpunkt für die Erfassung von Meßwerten ist, um das Intervall T_1 verschoben (siehe Bild 5.10 b)). Um das zu vermeiden wird hier die Vereinigung bereits im Steuereingriff j_{max} des laufenden Ausführungszeitraums berücksichtigt, wodurch auch die Ausführungstabelle geändert werden muß. Dies ist entgegen der Aussage von Abschnitt 5.2.1 ausnahmsweise zulässig,

da am Ende eines Zeitraums normalerweise keine Umschaltung stattfindet oder zu diesem Zeitpunkt alle Umschaltungen bereits ausgeführt sind. Die Behandlung der Zustandsänderung $\Delta z_{j_{max}}$ der Ausführungstabelle erfolgt wie oben beschrieben ebenfalls mit der XOR-Funktion. Findet zum Ende eines Zeitraums trotzdem eine Umschaltung statt, z.B. weil der letzte Steuereingriff j_{max} bereits einen vereinigten Steuereingriff darstellt, so hätte dies keine nachteiligen Auswirkungen. Im Fall, daß dort dieselbe Phase geschaltet wird, ergibt sich aufgrund der gleichen Zustandsänderung sogar eine Aufhebung der beiden Phasenumschaltungen.

Damit kann nun die komplette Berechnungsvorschrift für die Vereinigungsmethode angegeben werden, wobei der Index i die Ausführungstabelle des aktuellen Zeitraums bezeichnet und der Index $i+1$ die in diesem Zeitraum bearbeitete Berechnungstabelle für den darauffolgenden Zeitraum:

$$\left. \begin{aligned} \Delta T_{i+1,0}' &= \Delta T_{i+1,0} + \Delta T_{i+1,1} \\ \Delta z_{i,j_{max}}' &= \Delta z_{i,j_{max}} \text{ XOR } \Delta z_{i+1,1} \end{aligned} \right\} j = 1$$

$$\left. \begin{aligned} \Delta T_{i+1,j-1}' &= \Delta T_{i+1,j-1} + \Delta T_{i+1,j} \\ \Delta z_{i+1,j-1}' &= \Delta z_{i+1,j-1} \text{ XOR } \Delta z_{i+1,j} \end{aligned} \right\} j = 2, 3, \dots, j_{max}' \quad (5.9)$$

Allerdings treten durch diese Methode neue Arten von Steuereingriffen auf, z.B. wenn ein Steuereingriff der Art „Phasenumschaltung“ mit einem Eingriff der Art „Messung“ vereinigt wird.

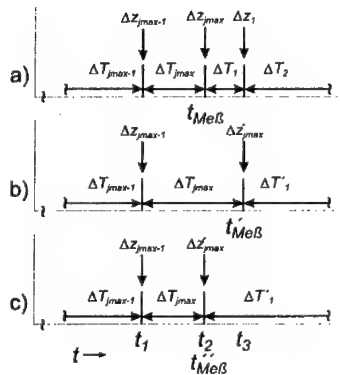


Bild 5.10: Spezielle Behandlung des Falls $\Delta T_1 < T_{min}$

Im Prinzip kann das Entscheidungskriterium einer Vereinigung nach Gleichung (5.5) noch genauer angegeben werden. In Bild 5.11 a) sind zwei Fälle dargestellt, für die zum einen $T_F < T_{min}/2$ (t_1 bis t_5) gilt und zum anderen $T_F \geq T_{min}/2$ (t_6 bis t_{10}). Die Mindestzeiten seien für alle Interrupts gleich. Die grau schattierten Blöcke in Bild 11 a) stellen die Lage der Interrupts dar, wenn keine Vereinigung stattfinden würde. Der Verlauf der zum Interrupt 1 bzw. 3 gehörenden Phase ist in Bild b) und c) grau dargestellt. Würde man nun in beiden Fällen das Kriterium

nach (5.5) anwenden, so ergäbe sich ein Verlauf nach Bild 5.11 b). Es ist zu erkennen, daß im Fall $T_F < T_{min}/2$ ein vergleichsweise großer Fehler der Spannungszeitfläche entstehen würde. Bild 5.11 c) zeigt die resultierenden Phasenspannungen, wenn statt (5.5) die Bedingung

$$T_{i,j} < \frac{T_{min}}{2}, \quad j = 1, 2, \dots, j_{max} \quad (5.10)$$

angewandt wird. Der Fehler der Spannungszeitfläche ist hier kleiner. Für $T_F \geq T_{min}/2$ ist der Fehler nach (5.5) identisch mit dem nach (5.11). Für die Fälle $T_F < T_{min}/2$ ergeben sich jetzt kleinere Fehler¹.

5.5.2.2 Warteschleifenmethode

Diese Methode verwendet zusätzlich zum Zeitwerk eine Warteschleife. Wenn zwei oder mehrere Steuereingriffe kurz hintereinander auftreten (siehe a) in Bild 5.12), wird der erste nach wie vor durch einen Interrupt ausgelöst. Der zweite Interrupt tritt aber innerhalb der Mindestzeit T_{min} nach Abschnitt 5.5.2.1 auf. Er kann deshalb nicht zum Zeitpunkt t_2 , sondern frühestens zum Zeitpunkt t_3 , entsprechend b) in Bild 5.12 ausgeführt werden. Treten mehrere Interrupts hinter-

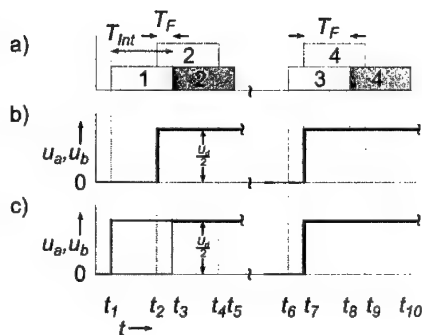


Bild 5.11: Verlauf zweier Phasenspannungen nach dem Kriterium (5.10) und (5.13).

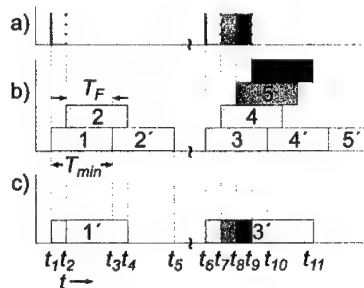


Bild 5.12: Darstellung der Warteschleifenmethode und deren Auswirkung auf die Laufzeit von Interruptroutinen

¹Dieser Vorschlag und weitere Ideen zur Verbesserung der Vereinigungsmethode wurden aber nicht weiter verfolgt, da die Warteschleifenmethode inzwischen auch realisiert war und sich als bedeutend besser und aussichtsreicher erwies.

einander auf (z.B. t_6, t_7, t_8, t_9), dann ergeben sich immer größere Fehlerzeiten. Mit der Warteschleifenmethode wird nur der erste Steuereingriff (t_1 bzw. t_6) durch einen Interrupt ausgelöst. Die nachfolgenden Steuereingriffe (t_2 bzw. t_7, t_8, t_9) werden innerhalb desselben Steuereingriffs nach dem Ablauf entsprechender Warteschleifen bearbeitet (siehe c) in Bild 5.12).

Zur Realisierung der Warteschleifenmethode muß die Interruptroutine, wie sie aus Abschnitt 5.3 bekannt ist, entsprechend Bild 5.13 erweitert werden. Zunächst wird der zugehörige Steuereingriff j ausgeführt und der nächste Zustand $j+1$ vorbereitet (siehe Bild 5.13). Als nächstes müßte jedoch das Steuereingriff-Intervall $j+1$ in das Zeitwerk geladen werden. Wenn jedoch die Gleichung (5.5) erfüllt ist (grau schattiert), übernimmt jetzt eine Sonderroutine die Steuerung. Sie enthält die eingriffsspezifischen Unterprogramme und alle anderen Anweisungen, die auch im Normalfall - ausschließlich des Ladens des Zeitwerks - abgearbeitet werden. So wird z.B. am Ende des Ausführungszeitraums die Messung gestartet und die Ausführungstabelle erneuert. Dann wird das folgende (kurze) Steuereingriff-Intervall mittels einer Warteschleife realisiert. Dabei muß berücksichtigt werden, daß die Einträge der Ausführungstabelle bereits eine Kompensation erster Art enthalten. Da aber keine neue Interruptroutine

ausgeführt wird, muß die Zeit der Warteschleife auf den unkompenzierten Wert korrigiert werden, indem ΔT_{j+1} wieder um T_{int} vergrößert wird. Nach Beendigung der Sonderroutine erfolgt ein Sprung an den Beginn der Interruptroutine selbst. Dort wird der Steuereingriff ausgeführt. Da mehrere kurze Intervalle aufeinander folgen können, wird die Abfrage des Kriteriums (5.5) wieder durchlaufen. So können beliebig viele Warteschleifen hintereinander stattfinden. Bei einem Steuereingriff am Ende des Ausführungszeitraums wird durch das eingriffsspezifische Unterprogramm die neue Ausführungstabelle geladen. So kann die Warteschleifenmethode

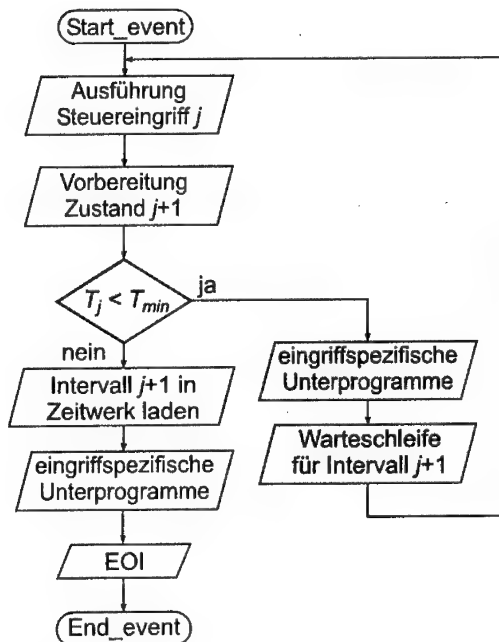


Bild 5.13: Modifizierte Interruptroutine für die Verwendung der Warteschleifenmethode

ohne besondere Zusatzmaßnahmen über die Grenze des Ausführungszeitraums hinaus ausgedehnt werden.

Die Warteschleife selbst besteht aus einer Schleife (Typ "for-to"), die keine Anweisung enthält. Die Zahl l der Schleifendurchläufe wurde als Variable von Typ `int` (=Integer) festgelegt. Damit lassen sich Wartezeiten bis zu 400 μ s realisieren.

Untersuchungen der Warteschleife selbst erfolgten mit dem in Bild 5.14 dargestellten Programmflußplan. Da die Systemuhr des PC nur mit einer Auflösung von 1/18.2 s abgelesen werden kann, müssen Zeitmessungen von kurzen Routinen durch vielfaches Wiederholen durchgeführt werden. Hierzu wurde eine überlagerte Schleife (Laufvariable m) mit konstantem Endwert (10^8 Durchläufe¹) verwendet. Unmittelbar vor Beginn der Wiederholungsschleife und nach deren Ende wird die Systemzeit ausgelesen (t_1 und t_2) und daraus die Laufzeit Δt der 10^8 -maligen Wiederholung der Warteschleife ermittelt. Die Auswertung der Untersuchung soll Aufschluß darüber bringen, welche Abhängigkeit zwischen der gemessenen Laufzeit einer Warteschleife und deren Endwert besteht, aber auch Hinweise auf die Ein- und Aussprunzeit der Zählschleife liefern. Ebenso läßt das Ergebnis auch Rückschlüsse über mögliche Nichtlinearitäten zu, die durch Sondereinflüsse verursacht werden könnten.

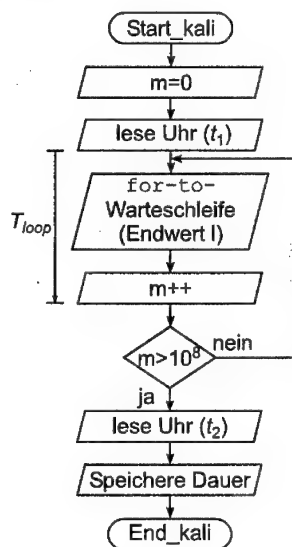


Bild 5.14: Programmflußplan zur Ermittlung der Warteschleifenparameter

Bild 5.15 zeigt das Ergebnis der Messung. Im Bereich sehr kleiner Werte von l ist eine Nichtlinearität zu erkennen, die näherungsweise wie eine Totzeit wirkt. Das kann auf mehrere Faktoren zurückgeführt werden. Zum einen muß auch bei Zählerendwert $l = 0$ die Schleife erst aufgebaut und das Abbruchkriterium überprüft werden. Außerdem nimmt das Auslesen der Systemzeit ebenfalls Zeit in Anspruch. Das erfolgt mit Hilfe eines Softwareinterrupts. Ähnlich wie Hardwareinterrupts benötigt auch diese Art eine vergleichsweise lange Zeit zur Ausführung (ca. 25 μ s). Schließlich muß noch die Wiederholschleife berücksichtigt werden, die bei

¹Diese Zahl gilt für das PentiumII-Systeme mit 400 MHz Taktfrequenz. Für andere Systeme muß dieser Wert ggf. angepaßt werden, damit die Laufzeit der Routine nicht zu groß oder zu klein wird.

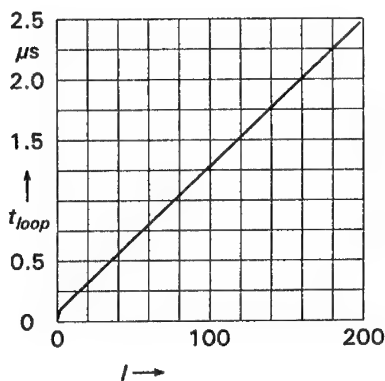


Bild 5.15: Gemessene Dauer der Warteschleife als Funktion des Schleifenendwerts l

kleinen Endwerten der Warteschleife l einen ähnlich hohen Aufwand an Rechenleistung benötigt.

Der Verlauf der Funktion ist etwa für Werte ab $l = 4$ nahezu linear. Das liegt darin begründet, daß - sobald die Zählschleife einmal installiert ist - jede weitere Erhöhung der Laufvariablen der Warteschleife eine konstante Zeit benötigt.

Damit eine auf den Rechner zugeschnittene Umrechnung von Wartezeiten in Warteschleifendurchläufe erfolgen kann, wird in einer Kalibrierungsroutine vor dem eigentlichen Stromrichterbetrieb die Funktion des

Bildes 5.15 gemessen. Die Kurve wird in guter Näherung als Gerade angenommen. Die Bestimmung der Parameter a und b der Geradengleichung

$$l = f(t) = a t + b \quad (5.11)$$

reduziert sich dadurch auf zwei Messungen mit einem kleinen Endwert ($l_1 = 4$) mit einem großen Endwert ($l_2 = 200$), welche die Wertepaare $(\Delta t_1, l_1)$ und $(\Delta t_2, l_2)$ liefern. Die Bestimmungsgleichung für die Parameter a und b ergibt sich damit zu

$$a = \frac{l_2 - l_1}{\Delta t_2 - \Delta t_1} \quad (5.12)$$

$$b = l_1 - a \Delta t_1$$

Diese Geradengleichung gibt Aufschluß darüber, welcher Schleifenendwert l notwendig ist, um eine Zeitdauer t verstreichen zu lassen. Sie vernachlässigt zwar die Nichtlinearität im Bereich kleiner Werte von l ($l < 4$), der resultierende Fehler ist jedoch tolerabel. Allerdings können rechnerisch auch negative Werte von l entstehen. Da es nicht sinnvoll ist negative Warteschleifen zuzulassen - zumal die Ausführung einer solchen länger dauert als die Ausführung einer Warteschleife mit dem Endwert $l = 0$ - werden deren Endwerte auf $l = 0$ korrigiert. Der dadurch entstehende Fehler (T_{Fehler}) wird dabei nicht größer als die Zeit T_{wait0} , die benötigt wird, um eine Warteschleife mit $l = 0$ auszuführen.

Da die Ausführungsgeschwindigkeit der Schleife stark hardwareabhängig ist, repräsentieren die Parameter a und b damit auch die Leistungsfähigkeit des Systems. Gerade deshalb ist der

Ermittlungsalgorithmus als solcher hardwareunabhängig und kann damit auf jedem anderen PC eingesetzt werden.

Es muß hier allerdings die Laufzeit der eingriffsspezifischen Unterprogramme berücksichtigt werden. Wie in Bild 5.13 dargestellt ist, wirkt die Laufzeit des Unterprogramms als zusätzlicher Teil des Warteintervalls. Die Mindestdauer T_{min} muß dies berücksichtigen. Man benötigt also den Betrag der Laufzeit des spezifischen Unterprogramms, damit die Warteschleife korrekt eingesetzt und berechnet werden kann. Die Messung der Laufzeit der einzelnen eingriffsspezifischen Unterprogramme erfolgt off-line noch vor dem eigentlichen Stromrichterbetrieb nach dem Wiederholungsprinzip und wird bei jedem Start der Software durchlaufen. Das Ergebnis der Messung wird als zusätzliche Information für jeden Steuereingriff in die Berechnungs- bzw. Ausführungstabelle aufgenommen (Element float t_{art}) und ist somit den spezifischen Eingriffen fest zugeordnet.

Unter der Berücksichtigung der Laufzeit der eingriffsspezifischen Unterprogramme sowie des Fehlers, der sich aus negativen Warteschleifen ergibt, kann nun T_{min} bzw. die tatsächliche Dauer der Warteschleife T_{wart} zu

$$\begin{aligned} T_{min} &= T_{art,j} + T_{int} + T_{Aus} + T_{Fehler} \\ T_{wart} &= \Delta T_j - T_{art,j} - T_{Aus} - T_{Fehler} \end{aligned} \quad (5.13)$$

angegeben werden. Sollte sich wiederum eine negative Warteschleife ergeben, d.h. $l < 0$, so errechnet sich der neue verbleibende Fehler T'_{Fehler} zu

$$T'_{Fehler} = -T_{wart} + T_{wart0} \quad (5.14)$$

und wird beim nächsten korrekten Ausstieg aus der Interruptroutine, d.h. $\Delta T_j > T_{min}$, berücksichtigt, so daß das für das Zeitwerk vorgesehene Intervall ΔT_j zusammen mit T_{Aus} den Wert

$$\Delta T'_j = \Delta T_j - T_{Aus} - T'_{Fehler}$$

erhält.

Der Vorteil dieser Methode ist, daß jedes beliebige Intervall nach Maßgabe der Zählschleifengenauigkeit (± 1) nachgebildet werden kann und so der tatsächliche Steuereingriff-Zeitpunkt erreicht wird. Somit existiert - bis auf den angegebenen Fehler bei sehr kleinen Zeiten - kein Unterschied zur berechneten, gewünschten Abfolge von Steuereingriffen. Die Abfrage des entscheidenden Kriteriums (5.5) erfolgt bei dieser Methode nicht zum Berechnungszeitpunkt, sondern immer zum Ausführungszeitpunkt. Wie oben beschrieben, kann damit sogar das Ende eines Zeitraums komplett mit der Warteschleifenmethode abgearbeitet werden. Da die Ausführungstabelle dort nicht modifiziert werden muß, finden demzufolge auch keine ggf. kritischen Eingriffe in der Ausführungstabelle statt.

5.5.2.3 Vergleich beider Verfahren

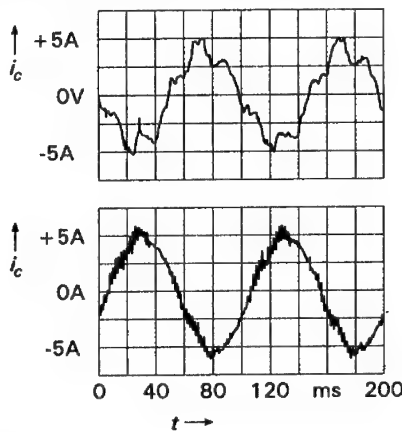


Bild 5.16:
Stromverläufe i_c bei Vereinigungsmethode
($U_d = 540$ V, $f = 10$ Hz); oben: $f_p = 5$ kHz;
unten: $f_p = 1,5$ kHz

Beide Verfahren wurden realisiert und experimentell untersucht. Jedes bietet eine Verringerung des Fehlers zweiter Art, bei der Vereinigungsmethode bleibt jedoch ein Restfehler bestehen. Die Restfehler treten vor allem bei kleinen Aussteuerungen auf, die bei Antrieben zusammen mit kleinen Ausgangsfrequenzen auftreten. Die Fehler verursachen dort in der Maschine erhöhte Verzerrungsströme. Das ist bei niedrigeren Pulsfrequenzen ($f_p < \text{ca. } 2 \text{ kHz}$) weniger kritisch, da hier der Restfehler bezogen auf eine Pulsperiode kleiner ist (siehe Bild 5.16). Das obere Bild zeigt den Stromverlauf bei einer Pulsfrequenz von 5 kHz und einer Ausgangsfrequenz von 10 Hz. Das untere Bild zeigt die gleiche Ausgangsfrequenz bei niedrigerer Pulsfrequenz (1,5

kHz). Es ist zu erkennen, daß die Verzerrungsströme deutlich niedriger sind als im oberen Bild. Deshalb kann die Vereinigungsmethode vor allem hier eingesetzt werden. Bei hohen Taktfrequenzen ($f_p > 5 \text{ kHz}$) werden der relative Restfehler und damit die Verzerrungsströme jedoch immer größer, so daß sie in diesem Bereich nicht geeignet ist.

Der Vorteil der Vereinigungsmethode ist, daß außer der Veränderung des Kriteriums für das Ende eines Zeitraums (5.8) die Interruptroutine nach Bild 5.3 nicht modifiziert zu werden braucht. Der Berechnungszeitpunkt der Kompensation nach der Vereinigungsmethode liegt außerhalb der Interruptroutine und daher nicht im zeitkritischen Bereich innerhalb der Interruptroutine selbst.

Die Warteschleifenmethode eliminiert im Gegensatz dazu praktisch vollständig Fehler zweiter Art. Dadurch kann sie auch bei höheren Pulsfrequenzen verwendet werden. Allerdings ist hier eine (fast) komplette Interruptroutine für den Sonderfall zu programmieren. Die Entscheidung, ob ein Sonderfall vorliegt, wird in der Interruptroutine selbst getroffen, so daß Eingriffe in den Arbeitsdatensatz entfallen. Dies hat jedoch zur Folge, daß die vorhergehende Kompensation erster Art rückgängig gemacht werden muß.

Die sinnvolle Einsatzmöglichkeit der Warteschleifenmethode über den gesamten Bereich der realisierbaren Pulsfrequenzen hinweg (siehe Bild 5.17) ist ein wesentlicher Vorteil. Deshalb wird im weiteren nur noch diese Maßnahme zur Vermeidung von Verzögerungen zweiter Art verfolgt.

Ein wichtiger Gesichtspunkt ist aber, daß beide Methoden nicht isoliert betrachtet werden brauchen. Aufgrund des unterschiedlichen Zeitpunkts des Einwirkens der beiden Methoden könnte es in Spezialfällen durchaus sinnvoll sein, ggf. Teile der Vereinigungsmethode in die Warteschleifenmethode zu integrieren. So können Ungenauigkeiten, die durch die Totzeit von Ein-/Ausgabebefehlen entstehen, weiter reduziert werden oder komplexe Pulsverfahren vereinfacht werden.

5.5.3 Umschaltunterdrückung

Aus unterschiedlichen Gründen, wie z.B. wegen der Einführung einer Nullkomponente (siehe Abschnitt 4.3.6) oder der Anwendung der Vollaussteuerung, kann es erforderlich sein, eine oder mehrere Phasen in einer Halbperiode nicht zu schalten. Das ist insbesondere dann der Fall, wenn die Umschaltzeitpunkte der Phasen den Wert 0 bzw. T_{PH} erreichen oder rechnerisch sogar unter- bzw. überschreiten. Dies ist in c) und d) von Bild 4.12 deutlich zu erkennen. Da z.B. in d) zum Zeitpunkt t_i die Umschaltung vom Typ A der Phase c mit der Umschaltung derselben Phase der Halbperiode vom Typ B zusammenfällt, heben sich beide Umschaltungen sogar auf. Das Kriterium für eine Umschaltunterdrückung lautet also

$$T_i \leq 0 \text{ ODER } T_i \geq T_{PH}, \quad i = a, b, c \quad (5.16)$$

Es kommt durchaus vor, daß in einer Halbperiode nur zwei Umschaltungen ausgeführt werden oder nur eine bzw. sogar keine Umschaltung stattfindet. Vielfach ist es sinnvoll, die Um-

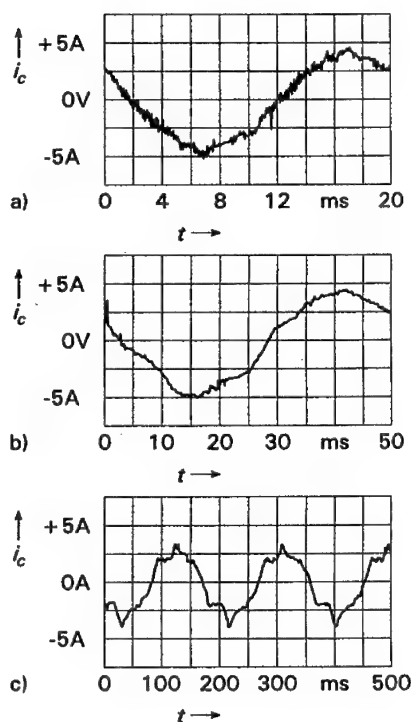


Bild 5.17: Stromverläufe bei Anwendung der Warteschleifenmethode
a) $U_d = 540 \text{ V}, f = 50 \text{ Hz}$
b) $U_d = 540 \text{ V}, f = 20 \text{ Hz}$
c) $U_d = 540 \text{ V}, f = 5 \text{ Hz}$

schaltunterdrückung schon eine kleine Zeitdauer ϵ ($\epsilon \approx 0.1 \mu\text{s}$) hinter bzw. vor dem Erreichen der Halbperiodengrenzen einzusetzen, d.h. $T_i \leq \epsilon$ bzw. $T_i \geq T_{pH} - \epsilon$. Das bedingt zwar eine Verkleinerung des Bereichs der realisierbaren Modulationsgrade, da der Wert von ϵ aber in der Größenordnung der realen Schaltzeiten der IGBT-Module liegt, könnten die Halbleiter so kleine Impulse nicht bilden. Die verwendeten IGBT-Ansteuermodule (siehe Abschnitt 3.3) sind bereits mit einer solchen Kurzimpuls-Unterdrückung ausgestattet, die Pulse einer Länge von weniger als $0.5 \mu\text{s}$ unterdrückt [5.3].

Für die Umschaltunterdrückung wird die Berechnungstabelle korrigiert. Es wird dadurch der richtige Zustand bei Beginn der Halbperiode eingestellt und der entsprechende Steuereingriff aus der Tabelle entfernt. Die Korrektur erfolgt nach dem zeitlichen Sortieren der Umschaltzeitpunkte aber vor dem Umrechnen in Umschaltintervalle, d.h. in der Tabelle stehen noch die Zeitpunkte ab Beginn der Halbperiode. Solche Tabellen werden im weiteren "vollständige Tabellen" genannt. An dieser Stelle sei nochmal daran erinnert, daß nach Abschnitt 5.2.1 der Steuereingriff für das Halbperiodenende stets der letzte Tabelleneintrag mit der Nummer j_{\max} ist, selbst wenn Phasenumschaltungen auftreten sollten, die zunächst größer als T_{pH} sind.

Die Richtung einer Umschaltung ist immer vom Typ der Halbperiode abhängig (siehe Abschnitt 4.1): In Halbperioden vom Typ A können Phasen nur von $-U_d/2$ nach $+U_d/2$ geschaltet werden und in Halbperioden vom Typ B nur umgekehrt. Damit die folgenden drei Phasenumschaltungen korrekt entsprechend Bild 5.18 vorgenommen werden können, muß - z.B. bei Verwendung der Sinusmodulation der Einzelphasen nach Abschnitt 4.3.6.1 - zum Zeitpunkt $t = 0$ ein vom Typ der Halbperiode abhängiger, im weiteren Normalzustand genannter Zustand herrschen. In der Realisierung wird er als Konstante z_{norm} berücksichtigt, die in sechsstelliger Notation den Wert

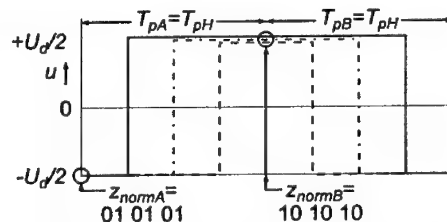


Bild 5.18: Normalzustände zu Beginn der Halbperioden des Typs A und B

$$z_{\text{norm}} = \begin{cases} 01 \ 01 \ 01 & \text{für Typ A} \\ 10 \ 10 \ 10 & \text{für Typ B} \end{cases} \quad (5.17)$$

annimmt. Für diese "vollständige Modulation" werden die Ausführungs- und die Berechnungstabelle als vollständige Tabellen erstellt. Sie beinhalten also die Daten von vier Steuereingriffen ($SE_j, j = 1, 2, \dots, j_{\max}$), mit denen die drei Phasenumschaltungen und das Ende eines Ausführungs- bzw. Berechnungszeitraums dargestellt werden (siehe Tabelle 5.2).

Umschaltunterdrückungen können aber erfordern, daß zum Zeitpunkt $t = 0$ ein vom Normalzustand abweichender Zustand, der im weiteren Anfangszustand genannt wird, herrschen muß, damit die Halbperiode korrekt ablaufen kann. Dies wird dadurch realisiert, daß beim Interrupt am Anfang des Ausführungszeitraums (Zeitpunkt $t = 0$) der notwendige Anfangszustand tatsächlich an den Stromrichter ausgegeben wird und so - ggf. mit Umschaltungen - der geforderte Stromrichterzustand herbeigeführt wird. Damit erreicht man eine Entkopplung der Berechnungen für die einzelnen Halbperioden. Der jeweilige Anfangszustand ist in der Variablen z_{Anfang} gespeichert. Sie wird zunächst auf den Normalzustand nach Gleichung (5.17) gesetzt und dann entsprechend den Schaltzeiten der Tabelle korrigiert.

Für jede erfolgte Umschaltunterdrückung wird die Interrupt-Reduktionsvariable sw_redN inkrementiert, sodaß - ähnlich wie bei der Vereinigungsmethode - das Halbperiodenende bereits beim Steuereingriff mit der Nummer $j_{max} - sw_redN$ erreicht wird.

5.5.3.1 Umschaltunterdrückung am Anfang einer Halbperiode

Die Variable z_{Anfang} enthält zunächst den Normalzustand. Wenn der erste relative Umschaltzeitpunkt T_i der vollständigen Berechnungstabelle die Bedingung (5.16) erfüllt, dann wird der Teilwert von z_{Anfang} für die entsprechende Phase mittels der XOR-Funktion neu eingestellt, d.h. bei Halbperioden vom Typ A wird "01" zu "10" und bei Typ B wird "10" zu "01". Dadurch wird sichergestellt, daß die Halbperiode für diese Phase richtig beginnt. Danach wird der zugehörige Steuereingriff aus der Tabelle entfernt und sw_redN inkrementiert. Die verbleibenden Steuereingriffe rücken in der Berechnungstabelle um eine Position nach vorn.

5.5.3.2 Umschaltunterdrückung am Ende einer Halbperiode

Dieser Fall tritt ein, wenn gemäß Bedingung (5.16) der letzte Umschaltzeitpunkt T_i größer als die Halbperiodengrenze T_{ph} ist und damit die Stromrichterumschaltung nicht mehr innerhalb des aktuellen Berechnungszeitraums ablaufen könnte, sondern erst im nächsten Berechnungszeitraum. Da aber der dann geltende Anfangszustand von Hause aus auf den richtigen Zustand für diese Phase eingestellt wird (siehe vorheriger Abschnitt), bleibt dieser Umschaltzeitpunkt sowie dessen Stromrichter-Zustandsänderung unabhängig vom Zustand an der Halbperiodengrenze in der aktuellen Berechnungstabelle im weiteren unberücksichtigt. D.h. der vorletzte Eintrag in der Tabelle wird entfernt (im letzten steht das Halbperiodenende), der Steuereingriff mit der Nummer j_{max} rückt um eine Position nach vorn und die Interrupt-Reduktionsvariable sw_redN wird wieder inkrementiert.

5.5.3.3 Mehrfachunterdrückungen

Wie eingangs erwähnt, müssen sich Umschaltunterdrückungen nicht auf nur eine Phase je Halbperiode beschränken. Es können auch mehrere Unterdrückungen auftreten, wobei alle möglichen Kombinationen von $T_i \leq 0$ und $T_i \geq T_{pH}$ ($i = 1, 2, 3$) richtig und vollständig erfasst werden müssen. Die in den Abschnitten 5.5.3.1 und 5.5.3.2 beschriebenen Algorithmen sind selbstverständlich auch im Fall von Mehrfachunterdrückungen gültig, da jede einzelne Phase betrachtet und deren Zustand unabhängig von den beiden anderen Phasen korrigiert wird. Es ist dabei zu berücksichtigen, daß sich im Falle von $T_i \leq 0$ die Tabelle mit jeder Unterdrückung um einen Eintrag verringert und dadurch auch entsprechend weniger Einzelverschiebungen von Steuereingriffen in der Berechnungstabelle vorgenommen werden müssen. Bild 5.19 zeigt

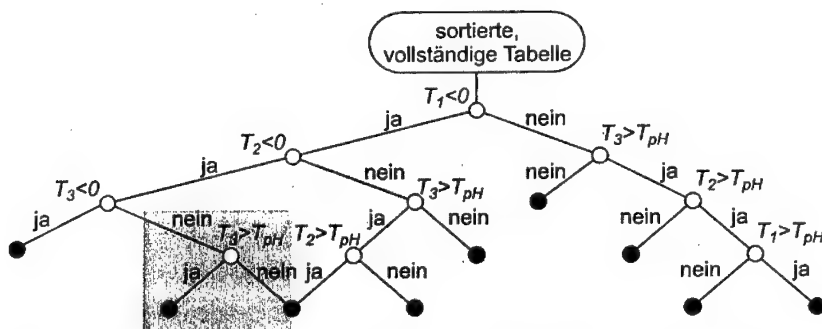


Bild 5.19: Reduzierter Entscheidungsbaum für die Unterdrückung von Umschaltungen

●...Endpunkt des Baums

○...Abfrage

den Entscheidungsbaum, welcher die möglichen, sinnvollen Kombinationen enthält. Er berücksichtigt dabei zwei Randbedingungen, die sich durch die Ordnung der drei Umschaltzeitpunkte in zeitlicher Reihenfolge ($T_1 < T_2 < T_3$) ergeben. Diese sind:

- (1) Wurde ein Umschaltzeitpunkt innerhalb der Halbperiode erkannt ($0 > T_i > T_{pH}$), so können alle zeitlich späteren Umschaltzeitpunkte nicht mehr im Bereich $T_i \leq 0$ liegen und brauchen daher nicht mehr untersucht zu werden.
- (2) Wurde ein Umschaltzeitpunkt innerhalb der Halbperiode erkannt ($0 > T_i > T_{pH}$), so können alle zeitlich früheren Umschaltzeitpunkte nicht mehr im Bereich $T_i \geq T_{pH}$ liegen und brauchen daher ebenfalls nicht mehr untersucht zu werden.

Die Indizes 1, 2, 3 in Bild 5.19 beziehen sich dabei auf die Reihenfolge der Umschaltzeitpunkte der vollständigen Tabelle. Da sich insbesondere bei Umschaltunterdrückungen am Anfang einer Halbperiode die Nummer der jeweiligen Steuereingriffe und damit deren Index innerhalb der Berechnungstabelle ändert, wird auch im folgenden Abschnitt auf die Bezeichnungen der vollständigen Tabelle bezuggenommen.

Im Prinzip kann dieses Verfahren auch auf zusätzlich eingeführte Steuereingriffe angewandt werden, die ebenfalls zu Umschaltungen von Transistoren führen wie z.B. zur Steuerung eines Brems-Choppers oder eines weiteren Stromrichters. Dadurch vergrößert sich zwar der Entscheidungsbaum, die beiden oben genannten Randbedingungen bleiben aber nach wie vor gültig.

5.5.3.4 Programmtechnische Realisierung

Für die rationelle Erfassung der vorliegenden Kombination, wird in der softwaretechnischen Realisierung für die Abfragen von $T_i < 0$ bei $i = 1$ begonnen und für die Abfragen von $T_i > T_{PH}$ bei $i = 3$, sodaß die beiden Randbedingungen, die zur Vereinfachung des Entscheidungsbaums in Bild 5.19 führten, einbezogen werden.

Zunächst werden alle Umschaltunterdrückungen zu Beginn eines Berechnungszeitraums entsprechend Bild 5.20 erfaßt. Dabei ist zu berücksichtigen, daß sich bei erfolgter Korrektur nach Gleichung Abschnitt 5.5.3.1 (= "Korrektur 1") bzw. 5.5.3.2 (= "Korrektur 2") der Tabellenindex der Steuereingriffe aufgrund des Reduzierens der Tabelle ändert, sodaß eigentlich immer der jeweils erste Eintrag der Tabelle untersucht wurde. Die Indizes in Bild 5.20 stimmen jedoch mit den Indizes der vollständigen Tabelle überein, wie es aus dem vorigen Abschnitt bekannt ist. Es ist zu erkennen, daß gemäß Randbedingung (1) abgebrochen wird sobald ein Umschaltzeitpunkt innerhalb des Berechnungszeitraums liegt.

Die Erfassung der Unterdrückungen am Ende des Berechnungszeitraums erfordert entsprechend des Entscheidungsbaums von Bild 5.19 für jede mögliche Anzahl an bereits unterdrückten Umschaltungen ($sw_redN = 0, 1, 2, 3$) eine eigene Abfrageroutine, die in Bild 5.20 entsprechend den Schattierungen des Ereignisbaums gekennzeichnet sind. Unabhängig vom Wert von sw_redN finden die Korrekturen hier immer wie in Abschnitt 5.5.3.2 beschrieben statt.

Wurden keine Umschaltungen zu Beginn des Berechnungszeitraums unterdrückt, so muß die Abfrageroutine prüfen, ob zumindest der größte Umschaltzeitpunkt T_3 außerhalb der Halbperiode liegt. Dies entspricht dem hellgrau schattierten Teil in Bild 5.19 und Bild 5.20. Durch eine Korrektur nach Abschnitt 5.5.3.2 fällt der letzte Tabelleneintrag weg, sodaß der nächste zu prüfende und zugleich größte verbleibende Umschaltzeitpunkt T_2 ist. Dies geht so fort bis ent-

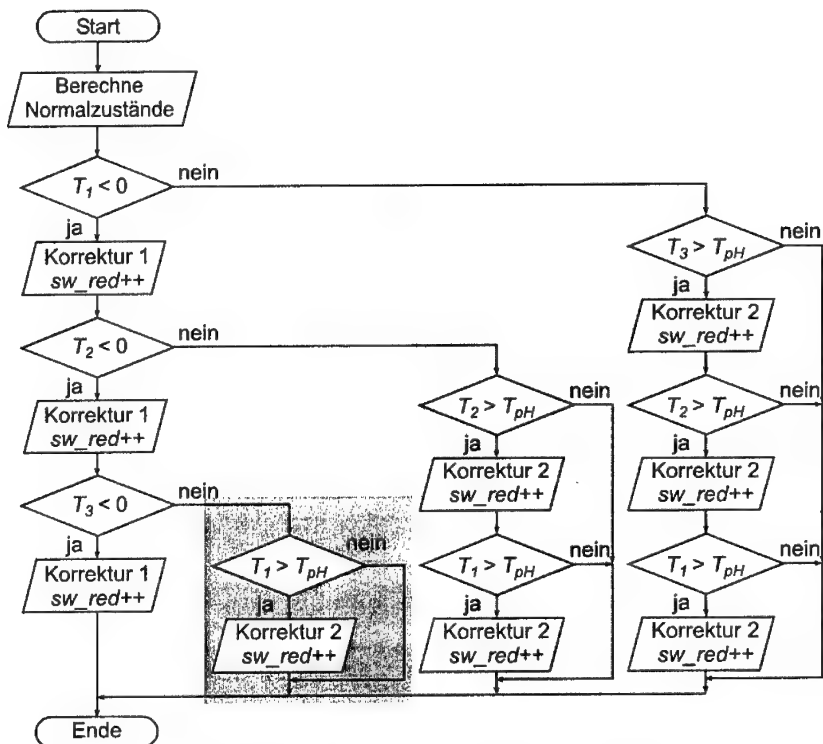


Bild 5.20: Programmflußplan für Mehrfachunterdrückungen nach Bild 5.19.

weder der letzte Umschaltzeitpunkt geprüft wurde oder die Routine aufgrund einer als falsch bewerteten Abfrage ($T_i \leq 0$) nach Randbedingung (1) abgebrochen wurde.

Falls bereits eine Umschaltunterdrückung zu Beginn des Berechnungszeitraums erfolgt ist, so ist der verbleibende, größte Umschaltzeitpunkt der Steuereingriff mit dem Index $j_{max}-2$ in der vollständigen Tabelle. Dies entspricht dem jeweils mittelgrau schattierten Teil. Es müssen nur noch die beiden verbleibenden der ursprünglichen drei Umschaltzeitpunkte untersucht werden. Ein ähnliches Vorgehen ergibt sich für $sw_redN = 2$ oder 3 (dunkelgrau schattiert),

wobei der Fall $sw_redN = 3$ trivial ist¹, da schon alle Umschaltungen zu Beginn des Zeitraums stattgefunden haben und eine weitere Prüfung entfallen kann.

Für die Umschaltunterdrückung wurden mehrere verschiedene Prinzipien und Verfahren realisiert und praktisch erprobt. Das hier vorgestellte erwies sich als besonders wirkungsvoll und anwendbar auf alle untersuchten Pulsverfahren. Trotzdem ist es einfach zu realisieren. Die entscheidende Idee ist die Einführung eines vom Verlauf des vorherigen Zeitraums unabhängigen Anfangszustandes einer Halbperiode. Er entkoppelt zwei aufeinander folgende Zeiträume, wodurch sich die Auswirkungen der berechneten Umschaltzeitpunkte ausschließlich auf die zur Berechnung zugrunde gelegte Halbperiode beschränken. Eine Berücksichtigung von Ereignissen zeitlich früherer Halbperioden ist also nicht erforderlich, was besonders die programmtechnische Umsetzung sehr vereinfacht.

5.5.4 Ingangsetzen und Abschalten des Stromrichters

Der Ruhezustand des Stromrichters ist durch "00 00 00" in der bereits erwähnten sechsstelligen Notation definiert. Das Ingangsetzen besteht darin, während des ersten Ausführungszeitraums die Transistoren eines Phasenschalters in zueinander inverse Zustände zu bringen. Andernfalls würden sich durch die direkte Anwendung der XOR-Verknüpfung zwischen dem aktuellen Zustand ("00 00 00") und der beabsichtigten Zustandsänderung (z.B. "00 00 11") Brückenkurzschlüsse ergeben. Um dies zu vermeiden, werden die Zustandsänderungen Δz der Steuereingriffe vom Typ "Phasenumschaltung" zunächst als "00 00 01" für Phase a, "00 01 00" für Phase b und "01 00 00" für Phase c angegeben. Da die erstmalige Berechnung der Ausführungstabelle off-line erfolgt, können die Zustandsänderungen Δz für Phasenumschaltungen nach der Berechnung der Ausführungstabelle auf die im weiteren gültigen Werte "00 00 11" für Phase a, "00 11 00" für Phase b usw. gestellt werden.

Beim Abschalten des Stromrichters müssen alle Transistoren des Wechselrichters ausgeschaltet werden, damit alle drei Phasen des Motors auf keinem der Potentiale $\pm U_d/2$ liegen. Das kann zwar ggf. noch zu einer Energierückspeisung über die Dioden eines Phasenschalters führen. Im allgemeinen ist das aber der Zustand mit dem geringsten Energieaustausch. Läge eine der Phasen auf $\pm U_d/2$, würden in den einzelnen Wicklungen die entsprechenden Kurzschlußströme fließen.

¹Das ist gültig, wenn anfangs eine vollständige Tabelle mit 3 Umschaltungen vorgelegen hat. Existieren aber mehr als die dort festgelegten Umschaltzeitpunkte, so kann sw_redN ggf. auch höhere Werte annehmen.

5.6 Umsetzung der Pulsverfahren

Die bisherigen Abschnitte behandelten weitgehend die Verarbeitung und Ausgabe von Stromrichterumschaltungen sowie deren PC-spezifischen Fehlerquellen. Es wurde davon ausgegangen, daß die Zeitpunkte der Umschaltungen bereits vorliegen. Dieser Abschnitt geht nun auf die Berechnung derselben und damit auf die eigentliche Umsetzung der in Abschnitt 4.3.6 beschriebenen Pulsverfahren ein.

5.6.1 Sollwertraumzeiger und Berechnungstabelle

Entsprechend Bild 5.21 ist der Ausgangspunkt für die Ermittlung der Berechnungstabelle ein Sollwert-Raumzeiger der Spannung¹ \underline{u}_w^* , dessen Mittelwert im Berechnungszeitraum nachgebildet werden soll (vgl. Bild 4.8). Dadurch bildet die Rücktransformation den Raumzeiger auf die Mittelwerte der drei Phasenspannungen im Berechnungszeitraum ab. Sie erfolgt für Polarkoordinaten des Sollwertes entsprechend Gleichung (4.15) mit

$$\begin{aligned} u_{wa0}^* &= \operatorname{Re}(\underline{u}_w^*) = |\underline{u}_w^*| \cos \alpha \\ u_{wb0}^* &= \operatorname{Re}(\underline{a}^2 \underline{u}_w^*) = |\underline{u}_w^*| \cos(120^\circ - \alpha) \quad (5.18 \text{ a}) \\ u_{wc0}^* &= \operatorname{Re}(\underline{a} \underline{u}_w^*) = |\underline{u}_w^*| \cos(240^\circ - \alpha) \end{aligned}$$

wobei α der Winkel des Raumzeigers \underline{u}_w^* ist. Für kartesische Koordinaten erfolgt die Rücktransformation analog nach Gleichung (4.13 b) mit

$$\begin{pmatrix} u_{wa0}^* \\ u_{wb0}^* \\ u_{wc0}^* \end{pmatrix} = \begin{pmatrix} \cos 0 & \sin 0 \\ \cos(2\pi/3) & \sin(2\pi/3) \\ \cos(4\pi/3) & \sin(4\pi/3) \end{pmatrix} \begin{pmatrix} u_{w\alpha}^* \\ u_{w\beta}^* \end{pmatrix} \quad (5.18 \text{ b})$$

Um die Darstellung der weiteren Gleichungen zu vereinfachen, werden für jede Phase der entsprechende Modulationsgrad m_a, m_b, m_c nach Gleichung (4.19) eingeführt, d.h.

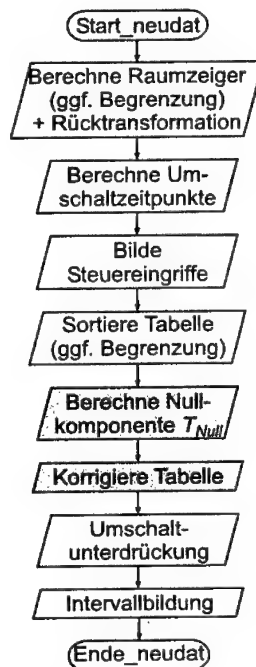


Bild 5.21:
Programmflußplan zur Umsetzung der Pulsverfahren

¹Auch bei der Stromregelung (siehe Abschnitt 5.7.1) wird ein Spannungsraumzeiger vorgegeben, da die Stellgröße des Stromrichters die Spannung ist.

$$m_i = \frac{u_{wi0}}{U_d/2}, \quad i = a, b, c \quad (5.19)$$

sodaß sich die (ungeordneten) Umschaltzeitpunkte T_i im Berechnungszeitraum zu

$$T_i = \begin{cases} \frac{T_{pA}}{2} (1 - m_i), & \text{Typ A} \\ \frac{T_{pB}}{2} (1 + m_i), & \text{Typ B} \end{cases} \quad i = a, b, c \quad (5.20)$$

ergeben. Dies entspricht genau Gleichung (4.23), wobei die für die unsortierte Berechnungstabelle nach Abschnitt 5.2.1 benutzten Bezeichnungen verwendet werden. Die Bezeichnung der Umschaltzeitpunkte enthält so keine Angaben mehr über den Typ der Halbperiode. Dadurch wird auch die Berechnungstabelle unabhängig vom Typ der Halbperiode, sodaß die Tabellen für jeden Halbperiodentyp einsetzbar sind.

Die Zeiten T_i werden zusammen mit den zugehörigen Stromrichter-Zustandsänderungen in den jeweiligen Steuereingriffen entsprechend Abschnitt 5.2.1 abgespeichert. Anschließend werden die Steuereingriffe ihrer zeitlichen Reihenfolge nach sortiert. Die Berechnungstabelle erhält dadurch eine Struktur gemäß Tabelle 5.2.

Bei Pulsverfahren, die eine Nullkomponente benutzen, wird an dieser Stelle die erforderliche Verschiebung T_{Null} der Umschaltzeitpunkte T berechnet und die Tabelle entsprechend korrigiert. Dies ist im folgenden Abschnitt näher beschrieben.

Die Tabelle wird dann auf ggf. durchzuführende Umschaltunterdrückungen nach Abschnitt 5.5.3 untersucht und entsprechend geändert. Auch bei Sinusmodulation der Einzelphasen kann eine Umschaltunterdrückung sinnvoll sein, da ggf. Fehler der Umschaltzeitpunkte aus früheren Halbperioden erst jetzt berücksichtigt werden können und zu Umschaltzeitpunkten außerhalb des aktuellen Berechnungszeitraums führen können. Abschließend wird die Intervallbildung nach Gleichung (5.1) durchgeführt. Dadurch ergibt die Tabelle 5.3. mit Intervallen ΔT_j .

5.6.2 Verfahren mit Nullkomponente

Die Darstellung und Realisierung von Pulsverfahren mit Nullkomponente über eine Verschiebung der Umschaltzeitpunkte fügt sich sehr gut in das vorgestellte Tabellen- und Berechnungskonzept ein. Besonders die Unterscheidung der Merkmale solcher Verfahren läßt sich auf diese Weise anschaulich herausarbeiten. Wie bereits angesprochen wird die Nullkomponente erst dann berücksichtigt, wenn die Umschaltzeitpunkte in geordneter Reihenfolge vor-

liegen. Dadurch ergeben sich wesentliche Vereinfachungen bezüglich der Berechnung der Nullkomponente. Alle vorhergehenden Berechnungsschritte sind aber für alle hier beschriebenen Pulsverfahren in gleicher Weise anwendbar.

Die Berechnung der Nullkomponente mit den Gleichungen (4.24), (4.26) bzw. (4.27) erfordert die Auswertung einer Minimal- und Maximalwertsuche hinsichtlich der Umschaltzeitpunkte. Das ist dann besonders einfach, wenn sie in zeitlicher Reihenfolge vorliegen, d.h. noch vor der Intervallbildung nach Gleichung (5.1) entsprechend den grau schattierten Anweisungen in Bild 5.21. Der Minimalwert der drei Umschaltzeitpunkte ist der Zeitpunkt des ersten Steuereingriffs (T_1) und der Maximalwert der des vorletzten (T_3)¹.

Die Berechnung der Verschiebung T_{Null} für dreiphasiges Pulsen mit symmetrischen Nullzuständen aus Abschnitt 4.3.6.2 vereinfacht sich also zu

$$T_{Null} = \frac{T_{pA,B} - T_3 - T_1}{2}, \quad \text{Typen A und B} \quad (5.21)$$

Für zweiphasiges Pulsen erster Art ergibt sich T_{Null} aus Abschnitt 4.3.6.3 zu

$$T_{Null} = \begin{cases} -T_1, & \text{Typ A, Einschalten +} \\ T_{pB} - T_3, & \text{Typ B, Einschalten -} \end{cases} \quad (5.22)$$

Schließlich ergibt sich T_{Null} für zweiphasiges Pulsen zweiter Art nach Abschnitt 4.3.6.4

$$T_{Null} = \begin{cases} T_{pA} - T_3, & \text{Typ A, Einschalten +} \\ -T_1, & \text{Typ B, Einschalten -} \end{cases} \quad (5.23)$$

Die Auswertung der drei Gleichungen liefert bereits vorzeichenrichtig den Wert T_{Null} , der allen Umschaltzeitpunkten hinzuaddiert wird, d.h.

$$T'_j = T_j + T_{Null}, \quad j = 1, 2, 3 \quad (5.24)$$

In der Realisierung findet die Vorauswahl des verwendeten Pulsverfahrens im Off-line-Teil vor dem eigentlichen Stromrichterbetrieb statt. Dort wird einer Variablen *nullkomp*, die vom Typ "Zeiger auf Funktion" ist, die entsprechende Routine zur Berechnung der Zeitdauer T_{Null} und der notwendigen Tabellenkorrektur zugewiesen. Der Variablen kann aber jederzeit auch die jeweilige Routine eines anderen Pulsverfahrens zugewiesen werden. Dadurch ist es bei Bedarf möglich, die Berechnungsvorschrift der Nullkomponente - und damit das Pulsverfahren selbst - in jeder Halbperiode zu wechseln. Im Programm selbst wird aber immer nur noch

¹Der letzte Steuereingriff beschreibt immer das Ende des Berechnungszeitraums.

die Funktion *nullkomp()* aufgerufen, sodaß eine aufwendige Abfrage nach dem aktuellen Pulsverfahren entfällt.

Bei Verwendung der Sinusmodulation der Einzelphasen muß *nullkomp* ebenfalls eine entsprechende Routine zugewiesen werden. Da bei diesem Pulsverfahren allerdings keine Nullkomponente auftritt, enthält diese Routine keine Anweisungen, sodaß sie nur aus dem Prozedurumpf ohne Anweisungen besteht. In der Programmiersprache C entspricht das "void name () { }". Die Funktionen für die drei Pulsverfahren mit Nullkomponente enthalten jeweils die zugehörige Gleichung (5.21), (5.22) oder (5.23) und immer die Anweisungen zur Korrektur der Umschaltzeitpunkte der Berechnungstabelle nach Gleichung (5.24).

Für Pulsverfahren, die sich nicht mit dem Programmflußplan nach Bild 5.21 realisieren lassen, könnte beispielsweise eine weitere Variable "Zeiger auf Funktion" deklariert werden, welche alle erforderlichen Anweisungen zur Umsetzung des Pulsverfahrens enthält. Im vorliegenden Fall würde die Funktion die gesamten Anweisungen des Programmflußplans nach Bild 5.21 enthalten. Auch hier kann in jeder Halbperiode der Variablen ein neuer Wert und damit der Berechnung eine andere Funktion zugewiesen werden.

Anmerkung: In der Literatur (z.B. [5.4], [5.5]) findet man häufig auch die Definition und die Beschreibung der verschiedenen Pulsverfahren über den Verlauf der Modulationsgrade m_a , m_b , m_c ¹ für die einzelnen Phasen. Dies führt beispielsweise zu einer einfachen Beschreibung des Oberschwingungsanteils und dessen Leistung. Die direkte Berechnung der Umschaltzeitpunkte aus dieser allgemein üblichen Art der Darstellung ist jedoch relativ aufwendig. Die hier gewählte Methode mit Ermittlung der Nullkomponente aus den Zeitpunkten der Sinusmodulation der Einzelphasen ist wesentlich einfacher.

5.6.3 Einphasige Pulsverfahren

Solche Verfahren bleiben in dieser Arbeit unberücksichtigt. Sie bilden entsprechend Abschnitt 4.3.6 einen Raumzeiger mit nur zwei Stromrichter-Spannungszuständen nach, sodaß höchstens eine Phase pro Halbperiode geschaltet wird. Meist kommen dabei nur sehr niedrige Pulsfrequenzen zum Einsatz. Dennoch ist es mit der beschriebenen Programm- und Tabellenstruktur möglich, auch solche Pulsverfahren zu realisieren.

¹Das gilt auch für Pulsverfahren mit Nullkomponente. Die Nullkomponente ist dann bereits im Modulationsgrad enthalten.

5.7 Regelung

Die Realisierung der Regelungen war nicht Aufgabe dieser Arbeit. Dennoch werden an dieser Stelle einige Aspekte beschrieben, welche die Verwirklichung innerhalb des beschriebenen

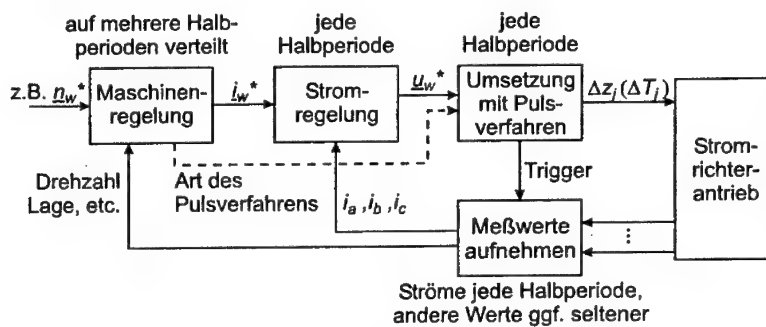


Bild 5.22: Typische Struktur eines Stromrichterantriebs mit Stromregelung und überlagerter Regelung

Konzeptes berücksichtigen. Bild 5.22 zeigt das Schema eines Stromrichterantriebs mit Regelung. Ausgehend von einem von außen vorgegebenen Sollwert - beispielsweise der Drehzahl - wird von der Maschinenregelung der Sollwert-Stromraumzeiger berechnet. Daraus wird vom Stromregler der Sollwert-Spannungsraumzeiger gebildet, der mit Hilfe eines ggf. von der Maschinenregelung vorgegebenen Pulsverfahrens umgesetzt wird. Die Aufnahme der Meßwerte (z.B. Ströme, Spannungen, Drehzahl usw.) des Stromrichterantriebs wird von der Steuerung veranlaßt und den entsprechenden Regelungen zugeführt. Die Häufigkeit des Aufrufs eines im Bild eingezeichneten Funktionsblocks (z.B. "Stromregelung") gibt einen Hinweis auf die Priorität, mit der er jeweils in der programmtechnischen Ausführung behandelt werden muß. Demzufolge muß durch die Programmstruktur gewährleistet sein, daß die Blöcke "Stromregelung", "Meßwerte aufnehmen" sowie "Umsetzung der Pulsverfahren" in jedem Berechnungszeitraum, d.h. in einer halben Pulsperiode, sicher durchlaufen werden. Das führt auf einen Programmflußplan, wie ihn Bild 5.23 zeigt. Er enthält die "Kernroutinen" des Berechnungszeitraums (unschattiert) und die Einflechtung von niedriger priorisierten Routinen. Das sind beispielsweise die Routinen der überlagerten Regelung bzw. der Kommunikation (hellgrau bzw. dunkelgrau schattiert). Dabei wird zwischen interruptfähigen Programmteilen und Programmteilen mit Mindestzeiten unterschieden. Interruptfähige Programmteile können innerhalb weniger Prozessortaktzyklen durch Interrupts unterbrochen werden. Sie dürfen daher ohne besondere Maßnahmen in eine Halbperiode eingebaut werden und sich über mehrere

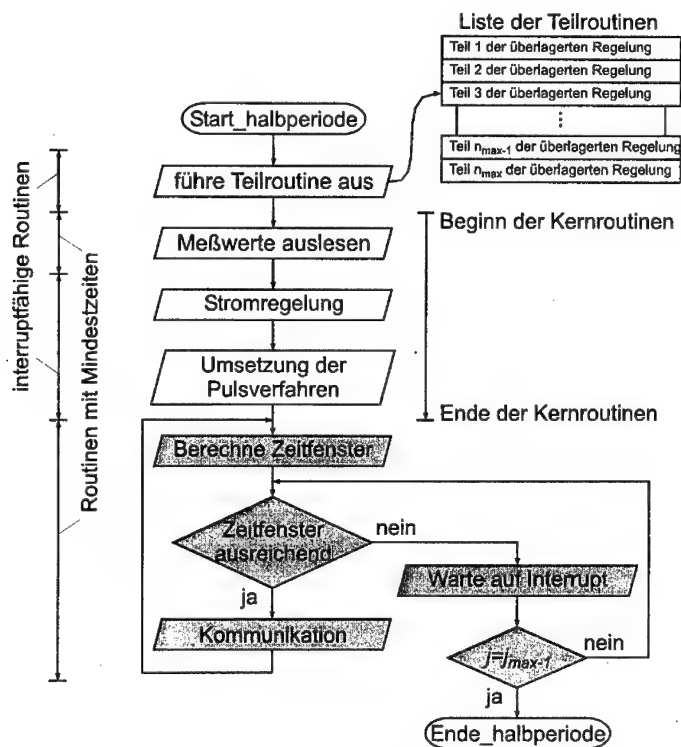


Bild 5.23: Programmstruktur bei Stromregelung und überlagelter Regelung

Steuereingriff-Intervalle hinweg verteilen. In den Routinen mit Mindestzeiten treten einzelne Befehle auf, die aufgrund ihrer eigenen Ausführungsdauer das Einleiten einer Interruptsequenz unzulässig lang blockieren (z.B. Ein-/Ausgabebefehle, siehe auch Abschnitt 5.4). Solche Routinen müssen in sogenannten Zeitfenstern ausgeführt werden, bei denen sicher ist, daß kein Interrupt des Zeitwerks auftritt. Dies ist dann der Fall, wenn das Zeitfenster innerhalb eines genügend großen Steuereingriff-Intervalls liegt. Die Unterscheidung der Programmteile bezüglich ihrer Interruptfähigkeit ist unabhängig von ihrer Priorisierung. Die folgenden Abschnitte gehen auf die einzelnen Teile der Bilder 5.22 und 5.23 näher ein.

5.7.1 Meßwerte

In Abschnitt 3.4.2.2 wurde erwähnt, daß die serielle Wandlung von analogen Meßwerten bei der verwendeten Meßkarte nur mit einer zeitlichen Verzögerung zwischen den einzelnen Ka-

nähen möglich ist. Wenn man alle Meßwerte gleichzeitig abtasten will, ist eine zusätzliche Schaltung erforderlich, die für jeden Kanal einen schnellen Sample&Hold-Baustein¹ enthält. Alle Bausteine werden dann gleichzeitig angesteuert, sodaß die Meßwerte auch alle dem gleichen Zeitpunkt entsprechen. Die Ansteuerung der Bausteine kann beispielsweise mit dem outp-Befehl erfolgen, der am Ende einer Halbperiode ausgegeben wird. Da der Wert des Stromrichterzustands nach Abschnitt 5.1.2 nur die niedrigsten sechs der acht Bit des outp-Befehls benötigt (z.B. "x₁x₂ 01 10 01"), kann eines der beiden verbleibenden Bits x₁ oder x₂ zur Ansteuerung aller Sample&Hold-Bausteine benutzt werden.

Die Zeitdauer einer A/D-Wandlung selbst bleibt dadurch aber unverändert. Bei der verwendeten Multi-I/O-Karte muß vor dem Auslesen der Meßwerte eine Wartezeit eingehalten werden, bis alle angeforderten Kanäle gewandelt sind². Erst dann dürfen die Werte ausgelesen werden, da sonst nur die von der Karte voreingestellten Werte übertragen werden, wodurch es zu falschen Werten kommt. Die Wartezeit kann auf zwei verschiedenen Wegen realisiert werden: Zum einen kann ein entsprechender Steuereingriff eingeführt werden, dessen einzige Aufgabe es ist, das Abbruchkriterium einer Warteschleife zu erfüllen. Zum anderen kann die Wartezeit dazu genutzt werden, um Teile der überlagerten Regelung entsprechend Bild 5.23 auszuführen. Dabei darf die Laufzeit der Teilroutinen die Wartezeit nicht unterschreiten, da es sonst zu den genannten Fehlern kommt.

Die Übertragung eines Meßwerts erfolgt durch das zweimalige Lesen der jeweiligen Basisadresse der Karte mit dem Befehl inp (Adresse). Er benötigt etwa die gleiche Zeit wie der Befehl outp nach Abschnitt 5.4.1 (ca. 0.6 µs bei PentiumII-Rechnern). Während dieser Zeit kann das System keine Hardware-Interrupt-Sequenz initiieren, sodaß z.B. ein unmittelbar nach einem inp-Befehl auftretender Interrupt erst mit einer Verzögerung von 0.6 µs bearbeitet wird³. Das Auslesen der Meßwerte ist also ein Programmteil mit Mindestzeit (siehe auch Bild 5.23), für den ein oder mehrere Zeitfenster zur Verfügung gestellt werden müssen. Aus dem Zeitbedarf der Teilroutinen der überlagerten Regelung (siehe Abschnitt 5.7.3), die im laufenden Ausführungszeitraum bearbeitet wurden, und der Steuereingriff-Intervalle ΔT_i der Ausführungstabelle läßt sich berechnen, wieviele Meßwerte bis zum nächsten Steuereingriff ausgelesen werden können. Solange nicht alle gewandelten Meßwerte ausgelesen sind, wird im

¹Es können beispielsweise Bausteine vom Typ AD781 verwendet werden.

²Bei der verwendeten Karte benötigt die Wandlung eines Kanals im "Burst-Modus" ca. 4 µs [5.6].

³Nach Abschnitt 2.3.2 wird bei Eintreffen einer Interrupt-Anforderung der aktuelle Befehl zuerst ausgeführt, bevor die Interrupt-Sequenz eingeleitet wird.

Steuereingriff selbst das nächste Zeitfenster neu berechnet. Im Prinzip führt diese Vorgehensweise auf eine Struktur ähnlich dem dunkelgrau schattierten Teil in Bild 5.23, wobei anstelle von "Kommunikation" die jeweiligen Meßwerte ausgelesen werden und zur Zeitfensterberechnung die Vorgehensweise aus Abschnitt 5.7.4.2 benutzt wird.

5.7.2 Stromregelung

Die Regelung der Phasenströme ist ohne Probleme mit der Halbperiodensteuerung realisierbar. Die erforderlichen aktuellen Ist-Werte werden dazu entsprechend Abschnitt 5.7.1 jeweils an den Halbperiodengrenzen gemessen. Nach Abschnitt 4.1 und 4.2 ist das zu diesen Zeitpunkten - zumindest theoretisch - ohne Oberschwingungsanteil möglich, sodaß die Fehler durch die Stromabweichung klein gehalten werden. Aus den aktuellen Momentanwerten der Phasenströme und dem Sollwert-Stromraumzeiger der überlagerten Regelung bildet der Stromregler einen Sollwert-Spannungsraumzeiger. Die Stromregelung muß - wie in Bild 5.23 dargestellt - zusammen mit den Routinen zur Umsetzung der Pulsverfahren unmittelbar nach der Meßwertaufnahme ausgeführt werden. Da in jeder Halbperiode neue Meßwerte vorliegen, kann die Routine für die Stromregelung ebenfalls in jeder Halbperiode aufgerufen werden. Die Verzögerung zwischen Messung und Stellsignal liegt damit im Bereich zwischen einer und zwei Halbperioden, sodaß sich ein sehr gutes Regelverhalten ergibt. Die durch die Stromregelung verursachte Mehrbelastung der CPU ist vergleichsweise gering und führt nicht zu ihrer Überlastung.

5.7.3 Überlagerte Regelung

In der Antriebstechnik enthält diese meist eine feldorientierte Maschinenregelung mit Drehzahl- und ggf. Lageregelung. Dazu müssen jedoch umfangreiche Maschinengleichungen ausgerechnet werden, was vergleichsweise zeitaufwendig ist. Da sich die Regelgröße hier im allgemeinen relativ langsam ändert, braucht die vorgesehene Routine nicht in jeder Halbperiode erneut ausgeführt werden, um eine befriedigende Ausregelzeit zu erzielen. Es reicht also aus, wenn die überlagerte Regelung erst nach mehreren Zyklen einen neuen Sollwert an die unterlagerte Stromregelung abgibt. Dadurch ist man in der Lage, die Rechenschritte der überlagerten Regelung entsprechend Bild 5.23 auf einen längeren Zeitraum zu verteilen. Solche Verfahren sind auch bei Verwendung von Microcontrollern durchaus üblich.

Zur Realisierung der Möglichkeit den kompletten Programmteil "überlagerte Regelung" auf mehrere Berechnungszeiträume aufteilen zu können, wird er in Teilroutinen aufgeteilt. Es ist vorteilhaft, sie in die Wartezeiten der Meßwertwandlung (siehe Abschnitt 5.7.1) einzubauen. Sie sind im allgemeinen ungleich lang, jedoch sollte ihre Laufzeit mindestens so groß wie die

Wartezeit sein. Dabei ist zu beachten, daß durch die Ausführung der Teilroutinen die Gesamtzeit der Halbperiode nicht überschritten wird. Der jeweilige Zeitbedarf einer Teilroutine wird ähnlich wie bei den Unterprogrammen für die Steuereingriffe im Off-line-Teil vor dem Stromrichterbetrieb ermittelt und zusammen mit den Teilroutinen in einer Liste abgespeichert, wobei der Zeitbedarf und die Routine selbst eine Zeile davon bilden. Mit Hilfe eines Indezzhählers, der mit jedem Aufruf der Liste erhöht wird, kann nun die entsprechende Teilroutine ausgeführt werden. Nach dem Durchlauf der Liste wird der Indezzhähler wieder auf Null zurückgesetzt, sodaß die überlagerte Regelung wieder vom Beginn starten kann.

Wie bereits erwähnt, findet die Ausführung der überlagerten Regelung während der Wartezeit der Messung (siehe Bild 5.23) - also unmittelbar nach dem Beginn des Berechnungszeitraums - statt. Obwohl sie nicht die höchste Priorität besitzt, ist es dennoch sinnvoll, Teile davon dort unterzubringen, da sie interruptfähig und dadurch im Prinzip beliebig verteilbar sind. Andernfalls würden ggf. Teile der Wartezeit durch die Zeitfensterbildung ungenutzt verstreichen. Ein weiterer Vorteil ist, daß genau festgelegt werden kann, auf wieviele Zeiträume sich die überlagerte Regelung verteilt. Falls die überlagerte Regelung mehrere Gruppen mit verschiedenen Zykluszeiten enthält, können einzelne Gruppen mit einer eigenen Liste an das Ende der Kernroutinen nach Bild 5.23 gesetzt werden. Für diese Teile ist dann eine genaue Ermittlung der Laufzeit nicht erforderlich. Es muß aber stets sichergestellt werden, daß die Halbperiode zum Ablauf ausreicht.

5.7.4 Zeitfenster

Wie in Abschnitt 6 beschrieben, erfolgt der Datenaustausch mit der Umgebung¹ weitgehend über Lese- bzw. Schreibzugriffe auf die Basisadressen der jeweiligen (externen) Geräte mit Hilfe von Ein-/Ausgabebefehlen. Die zugehörigen Routinen sind also nicht interruptfähig und müssen daher in Zeitfenstern ausgeführt werden. Im allgemeinen können Zeitfenster zu jedem beliebigen Zeitpunkt innerhalb des Ausführungszeitraums definiert werden. Da aber - mit Ausnahme der Meßwertaufnahme - in Zeitfenstern niedrig priorisierte Routinen ablaufen, ist es aus Gründen der Echtzeitfähigkeit sinnvoll, deren Bearbeitung an das Ende des Berechnungszeitraums zu legen.

Es wird zunächst geprüft, ob das berechnete Zeitfenster der Dauer T_{rest} groß genug ist, um eine bestimmte Routine mit Mindestzeit (z.B. eine Ein-/Ausgaberoutine) ausführen zu können. Ist das der Fall, so wird nach deren Bearbeitung die Restzeit um die Laufzeit der Routine verringert, d.h.

¹Das beinhaltet auch das Auslesen der Meßwerte.

$$T_{rest}' = T_{rest} - T_{lauf} \quad (5.25)$$

und erneut geprüft, ob die Größe der verbleibenden Restzeit T_{rest}' ausreicht, um eine weitere solche Routine auszuführen. Ergibt jedoch die Prüfung, daß der nächste Steuereingriff zur Laufzeit der vorgesehenen Routine auftritt, so wird statt dessen eine Warteschleife ausgeführt, die bis zum Eintritt des Steuereingriffs dauert. Im Steuereingriff selbst wird das Zeitfenster auf einen neuen Wert gesetzt, der sich aus dem in das Zeitwerk geladenen Steuereingriff-Intervall ΔT_{j+1} ergibt, das entsprechend dem grau schattierten Teil in Bild 5.24 noch zusätzlich um die Laufzeit des eingriffsspezifischen Unterprogramms $T_{zus,j}$ und der Aussprunzeit T_{aus} verringert wurde.

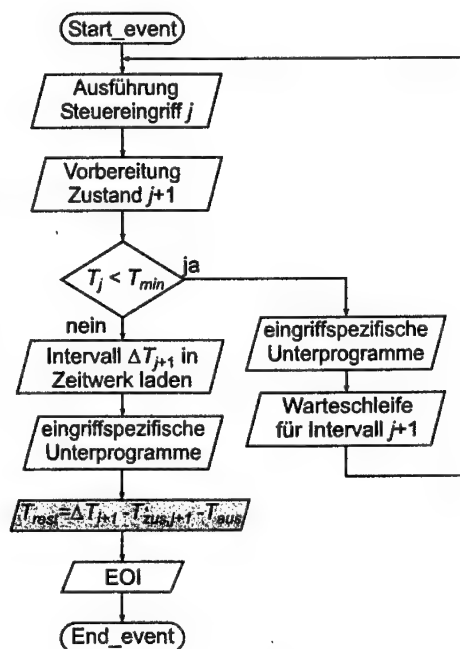


Bild 5.24: Modifizierter Programmflußplan der Steuereingriffe

Für die erstmalige Berechnung eines Zeitfensters innerhalb eines Ausführungszeitraums existieren mehrere Möglichkeiten. Sie sind immer in eine Struktur entsprechend dem dunkelgrau schattierten Teil in Bild 5.23 eingebettet und werden im weiteren vorgestellt.

5.7.4.1 Berechnung nach Steuereingriff

Die einfachste Methode zur Berechnung von Zeitfenstern besteht darin, nach Ende der Kernroutinen den nächsten Steuereingriff abzuwarten und dann erst das von ihm neu gesetzte Zeitfenster zu verwenden.

Wegen der Einfachheit sollte diese Methode stets eingebaut werden. Sie reicht aber allein nicht aus, denn bei einigen Aussteuerungen werden hiermit überhaupt keine Zeitfenster zur Verfügung gestellt werden können. Das ist insbesondere bei zweiphasigem Pulsen erster und zweiter Art in Verbindung mit kleinen Modulationsgraden der Fall. Die Steuereingriffe treten hier nur im Bereich der Halbperiodengrenzen auf, sodaß die Eingriffe entweder schon alle bearbeitet sind, bevor die Kernroutinen durchlaufen sind oder sie liegen so dicht am Ende des Ausführungszeitraums, daß kein Fenster mehr Platz findet, d.h. das Warten auf den nächsten

Steuereingriff führt entweder direkt zur Halbperiodengrenze oder zu Steuereingriffen in der Nähe davon. In beiden Fällen bleibt keine Möglichkeit mit dieser Methode ein Zeitfenster zu benennen und es müssen zusätzliche Methoden zur Ermittlung von Zeitfenstern in diesen Fällen benutzt werden.

5.7.4.2 Berechnung durch Auslesen des Zeitwerks

Die Grundlage dieses Verfahrens liegt darin, daß aus dem Zeitwerk auch der aktuelle Zählerwert $N_{\text{Zähler}}$ ausgelesen werden kann, sodaß sich über die Kenntnis der Taktfrequenz f_{Takt} des Zeitwerks und der Nummer des laufenden Steuereingriff-Intervalls j die aktuelle zeitliche Lage T_{akt} innerhalb des Intervalls berechnen läßt. Damit ist auch die in Intervall verbleibende Restzeit T_{rest} bekannt. Sie ergibt sich damit zu

$$T_{\text{rest}} = \Delta T_j - N_{\text{Zähler}} f_{\text{Takt}}, \quad j = 1, 2, \dots, j_{\text{max}} \quad (5.26)$$

und stellt das im laufenden Intervall größtmögliche Zeitfenster dar.

Der Vorteil dieser Methode liegt darin, ohne großen Aufwand sofort eine exakte Information über die Größe des Zeitfensters zu erhalten. Die Bereitstellung von Zeitfenstern kann damit ohne Zeitverzögerung erfolgen. Der Nachteil ist jedoch, daß nach [5.7] für das Auslesen des Zeitwerks Ein-/Ausgabebefehle notwendig sind, was zu einer möglichen Verzögerung von Interrupts führen kann. Dadurch wird die Berechnung des Zeitfensters selbst zu einer Routine mit Mindestzeit und müßte daher ebenfalls in einem Zeitfenster ausgeführt werden. Das führt auf eine Rekursion der Zeitfensterberechnung, die nicht ausführbar ist.

5.7.4.3 Berechnung durch Laufzeitmessung

Dieses Verfahren beruht darauf, daß die Laufzeit jedes einzelnen verwendeten Unterprogramms im Off-line-Teil meßbar ist. Daraus läßt sich vergleichsweise genau die im laufenden Ausführungszeitraum verstrichene Zeit T_{AZ} abschätzen, die sich in etwa zu

$$T_{\text{AZ}} \approx \sum_{n=0}^l T_n \leq T_p / 2 \quad (5.27)$$

ergibt, wobei T_n die Laufzeiten der bis zum Berechnungszeitpunkt des Fensters bearbeiteten Unterprogramme sind. Das Zeitfenster im laufenden Steuereingriff-Intervall ΔT_j berechnet sich so zu

$$T_{\text{rest}} = \sum_{n=0}^l \Delta T_n - T_{\text{AZ}} \quad (5.28)$$

sodaß diese Methode immer in der Lage ist, Zeitfenster zur Verfügung zu stellen. Die vorab beschriebenen Strukturen zur Überprüfung des Zeitfensters bzw. des erneuten Setzens des Fensters (dunkelgrau schattiert in Bild 5.23) gelten auch hier.

Im Prinzip kann diese Methode dadurch vereinfacht werden, indem man die Teilroutinen so gestaltet, daß sie alle oder teilweise kürzer sind als die vorgegebene Mindestzeit. Auf diese Weise entfallen die Laufzeitmessungen für die entsprechenden Teilroutinen.

Der Vorteil dieser Vorgehensweise ist, daß zur Bestimmung des zeitlichen Standorts innerhalb des Steuereingriff-Intervalls keine Ein-/Ausgabebefehle auftreten. Sie ist also interruptfähig. Der Nachteil liegt darin, daß ein verhältnismäßig großer Aufwand an Laufzeitmessungen im Off-Line-Teil notwendig ist, um alle Unterprogramme zu messen. Außerdem können sich Anzahl i_{max} und Art der Unterprogramme ggf. sogar von Halbperiode zu Halbperiode ändern, sodaß ein spezielles Protokollverfahren entwickelt werden muß, um alle wirklich bearbeiteten Unterprogramme zu erfassen. Ein weiterer Nachteil ist, daß auftretende Unsicherheiten in der Laufzeitmessung zu einer falschen Einschätzung der Zeitverhältnisse entsprechend den Gleichungen (5.27) und (5.28) führen würden und dann ggf. Zeitfenster zur Verfügung gestellt würden, in denen Steuereingriffe auftreten könnten.

5.7.5 Begrenzung des Sollwert-Spannungsraumzeigers

Bei Regelungen kann es vorkommen, daß die vom Regler ausgegebene Stellgröße den erlaubten Bereich überschreitet. Mit der Rücktransformation des Sollwert-Raumzeigers der Spannung entsprechend Abschnitt 5.6.1 können also Modulationsgrade der Phasen entstehen, deren Betrag den Wert 1 überschreiten. Solche Modulationsgrade führen bei der Berechnung zu Umschaltzeitpunkten, die außerhalb der Grenzen des Berechnungszeitraums liegen.

Durch die Überprüfung der Umschaltzeitpunkte hinsichtlich möglicher Umschaltunterdrückungen werden nach Abschnitt 5.5.3 die Phasenumschaltungen auf den Bereich des Berechnungszeitraums reduziert. Das kann bei zunehmendem Betrag des Raumzeigers dazu führen, daß keine Umschaltung mehr im Berechnungszeitraum auftritt bzw. nur dann, wenn die Reihenfolge der Phasenumschaltungen während eines Raumzeiger-Umlaufs (z.B. von a-b-c auf c-a-b) wechselt. Diese ist aber nur vom jeweiligen 60°-Sektor entsprechend Bild 4.6 abhängig, in dem sich der Raumzeiger befindet und das entspricht letztlich der Vollaussteuerung. Diese Art der Begrenzung betrifft nicht den Wert des Sollwert-Raumzeigers selbst, sondern ist die Folge der hier beschriebenen Stromrichtersteuerung¹, womit für beliebige Sollwertvorgaben ein störungsfreier Betrieb ermöglicht wird.

¹Da hier für die Begrenzung keine zusätzlichen Berechnungen notwendig sind, kann sie auch als "passive Begrenzung" bezeichnet werden.

Es kann auch eine weitere Möglichkeit der Raumzeigerbegrenzung eingebaut werden, wenn daraus zusätzliche Maßnahmen in der Steuerung oder Regelung abgeleitet werden sollen¹. Dafür wird der Raumzeiger aktiv so beschränkt, daß sich maximal der Betrag eines Raumzeigers ergibt, der noch innerhalb des Sechsecks der Spannungszustände liegt. Dabei geht man davon aus, daß die Aussteuerungen zweier Phasen eine Differenz von maximal 2 aufweisen können. Betrachtet man jeweils die Phase mit der größten und der kleinsten Aussteuerung, dann wird für diesen Fall eine Phase mit $m = +1$ und die andere mit $m = -1$ angesteuert. Über- oder unterschreitet nun die Differenz dieser Aussteuerungen den Wert 2, dann müssen alle drei Aussteuerungen mit dem Faktor

$$K = \frac{2}{\max(m_a, m_b, m_c) - \min(m_a, m_b, m_c)} \leq 1 \quad (5.29)$$

gemäß

$$m_{i,K} = K \cdot m_i, \quad i = a, b, c \quad (5.30)$$

korrigiert werden. Der Wert des Modulationsgrads m_i ist unabhängig vom Typ der Halbperiode und daher entspricht die weitere Umsetzung des Pulsverfahrens dem Bild 5.21.

Diese Art der Raumzeigerbegrenzung läßt sich sehr leicht auf die Umschaltzeitpunkte übertragen, sofern sie in zeitlich sortierter Reihenfolge vorliegen und noch keine Nullkomponente enthalten. Aus Gleichung (5.20), (5.29) und (5.30) ergibt sich demnach für K

$$K = \frac{T_{pH}}{T_3 - T_1} \leq 1 \quad (5.31)$$

sodaß sich die korrigierten Umschaltzeitpunkte T_j' zu

$$T_j' = T_{pH} - K(T_{pH} - T_j), \quad \text{Typen A und B} \quad (5.32)$$

berechnen. Hier ist die Unterscheidung der Halbperioden allerdings wichtig, da die Umschaltzeitpunkte über die entsprechende, typabhängige Form der Gleichung (5.20) berechnet wurden und dementsprechend korrigiert werden müssen. Anschließend wird die jeweilige Nullkomponente entsprechend den Gleichungen (5.21) - (5.23) berechnet und hinzuaddiert.

Bei beiden Varianten, d.h. bei Korrektur der Modulationsgrade bzw. der Umschaltzeiten, wird der ursprüngliche Raumzeiger so gestaucht, daß der resultierende Raumzeiger genau auf der Begrenzungslinie des Sechsecks liegt. Gleichzeitig bleibt der Winkel des Raumzeigers zur Realteil-Achse gleich. Es ist aber auch denkbar, daß der Raumzeiger auf einen noch kleineren

¹Bei der passiven Begrenzung ist das schwieriger zu realisieren.

Wert gestaucht wird. Dadurch würde sich ein anderer (kleinerer) Wert für K ergeben, die Gleichungen (5.30) bzw. (5.32) blieben davon aber unberührt.

6 Kommunikation

Trotz der vielfältigen Möglichkeiten eines PC-Systems bezüglich der Ein- und Ausgabe von Daten muß immer berücksichtigt werden, daß dies vergleichsweise zeitaufwendig ist. Die Nutzung der umfangreichen Ein- und Ausgabemöglichkeiten des AT-Modells kann deshalb einen Konflikt mit der Forderung nach Echtzeitverhalten des Systems bei höheren Pulsfrequenzen herbeiführen. Deshalb muß eine genaue Untersuchung über den Zeitbedarf der Routinen zur Bedienung der einzelnen Ein-/Ausgabegeräte vorausgehen, sodaß sie nur zu den Zeiten aufgerufen werden, zu denen keine Steuereingriffe zu erwarten sind und diese nicht behindern können. Die Übertragung von Daten kann ggf. mehrere Zeitfenster entsprechend Abschnitt 5.7.4 erfordern. Vor diesem Hintergrund ist klar, daß die übertragenen Daten - auch wenn sie Betriebsparameter beeinflussen - stochastische Verzögerungen aufweisen. Es ist aber durchaus denkbar, daß z.B. die (vergleichsweise langsame) Maschinenregelung ihre Sollwerte auf diese Weise von einem externen Rechner empfängt.

Es sei an dieser Stelle angemerkt, daß im allgemeinen innerhalb von Hardware-Interruptroutinen keine Software-Interrupts und damit z.B. auch keine Bildschirmausgaben ausgeführt werden dürfen. Dies führt zum unmittelbaren Absturz des Rechners, da DOS ein nicht reentrantes Betriebssystem ist, d.h. es kann sich nicht selbst aufrufen (siehe Literatur [6.1]). Alle Arten von Interrupts rufen jedoch Betriebssystemfunktionen auf und sind damit Bestandteil des Betriebssystems¹, weshalb ein erneuter Betriebssystemaufruf durch Software-Interrupts von dieser Ebene aus nicht möglich ist.

6.1 Tastatur

Die Tastatur als Bestandteil eines PC stellt ein wesentliches Instrument zur Dateneingabe dar. Obwohl der Tastaturcontroller (siehe Abschnitt 2.2.7) bidirektional arbeiten kann, d.h. er kann auch Daten an die Tastatur *ausgeben*, sind seine Möglichkeiten in dieser Hinsicht beschränkt, da eine Tastatur im AT-System eigentlich nur über drei Anzeige-LED verfügt², die den

¹Das gilt auch für selbst programmierte Interruptroutinen, die allerdings dann nur zeitweise Bestandteil des Betriebssystems sind.

²Das sind die Anzeigen für die Zustände "numerischer Block EIN/AUS", "Großbuchstaben EIN/AUS" sowie "Rollfunktion EIN/AUS"

Zustand der Tastatur anzeigen. Deshalb wird die Tastatur im weiteren nur noch als Eingabegerät verstanden.

Im Normalbetrieb veranlaßt der Tastaturcontroller bei Empfang eines Tastatursignals die zugehörige Interruptlogik zur Anforderung eines Hardwareinterrupts. Entsprechend Abschnitt 2.2.7 generiert der Tastaturcontroller bei jedem Drücken oder Loslassen einer Taste einen Make- bzw. Break-Code, der zunächst von den jeweiligen Interruptroutinen des Betriebssystems oder der Anwendung aus dem Tastaturcontroller ausgelesen und anschließend interpretiert wird. Auf diese Weise sind vielfältige Funktionen des Systems oder der jeweiligen Anwendung programmierbar, die auf einen einfachen Tastendruck oder eine Tastenkombination hin gestartet werden können.

Bei der Steuerung von Stromrichtern mit PC ist die interruptgesteuerte Dateneingabe über die Tastatur jedoch aufgrund der relativ langen Laufzeit der entsprechenden Interruptroutine wenig praktikabel. Da außerdem über die Tastatur meist nur binäre Informationen, d.h. Taste gedrückt bzw. nicht gedrückt, mit vergleichsweise niedriger Wiederholrate und damit keine reellen Werte eingegeben werden, ist es sinnvoll, den Tastaturinterrupt zu maskieren und die Daten über ein Pollingverfahren (vergleiche Abschnitt 2.3.2) aus dem Tastaturcontroller direkt auszulesen. Dazu braucht zunächst nur das Statusregister des Controllers (Basisadresse 0x64_h) ausgelesen zu werden. Nur wenn das Zustandsbit dieses Registers anzeigt, daß im Ausgabepuffer ein vollständiges Byte zum Auslesen bereit liegt (Bit 0 = 1), wird es ausgelesen und weiter verarbeitet. Das Auslesen selbst erfolgt bei der Programmiersprache C mit Hilfe des Befehls `inp (Basisadresse)`. Das ausgelesene Byte stellt den vom Tastaturcontroller ermittelten Scancode in Form eines vorzeichenlosen Ein-Byte-Wertes¹ dar.

In der vorliegenden Arbeit wurde auf diesem Wege eine Tastatursteuerung zur manuellen Änderung der Amplitude der Sollwertspannung (Modulationsgrad) bzw. des Sollwert-Spannungsraumzeigers sowie der Frequenz der Ausgangsspannung realisiert. Der Programmflußplan nach Bild 6.1 zeigt die Ausführung, wonach die Frequenz um 1 Hz vergrößert bzw. verkleinert wird, wenn die Taste "w" bzw. "s" gedrückt wurde oder der Modulationsgrad der Sollwertspannung um 0.01 erhöht bzw. verringert wird, wenn die Taste "q" bzw. "a" gedrückt wurde. Die Werte 30, 16, 31, 17 und 28 entsprechen den Make-Codes der Tasten "q", "a", "w", "s" sowie "Enter". Das Drücken der letztgenannten Taste führt zum Ende des Hauptprogramms. Der mit dem Loslassen einer Taste erzeugte Break-Code bleibt hier unberück-

¹In C entspricht das eigentlich dem Variablentyp "unsigned char" mit einem Wertebereich vom 0 bis 255. Es ist aber durchaus möglich den Scan-Code in einer Integer-Variablen abzuspeichern.

sichtigt. In Bild 6.1 ist deutlich zu sehen, daß Eingaben nach der beschriebenen Methode schnell zu umfangreichen, zeitaufwendigen Abfrageroutinen führen. Daher wurde die Anzahl der unterschiedlichen Eingabeinformationen im praktischen Einsatz auf wenige wichtige Funktionen, wie z.B. das Beenden des Programms oder den Wechsel zwischen verschiedenen Pulsverfahren, begrenzt.

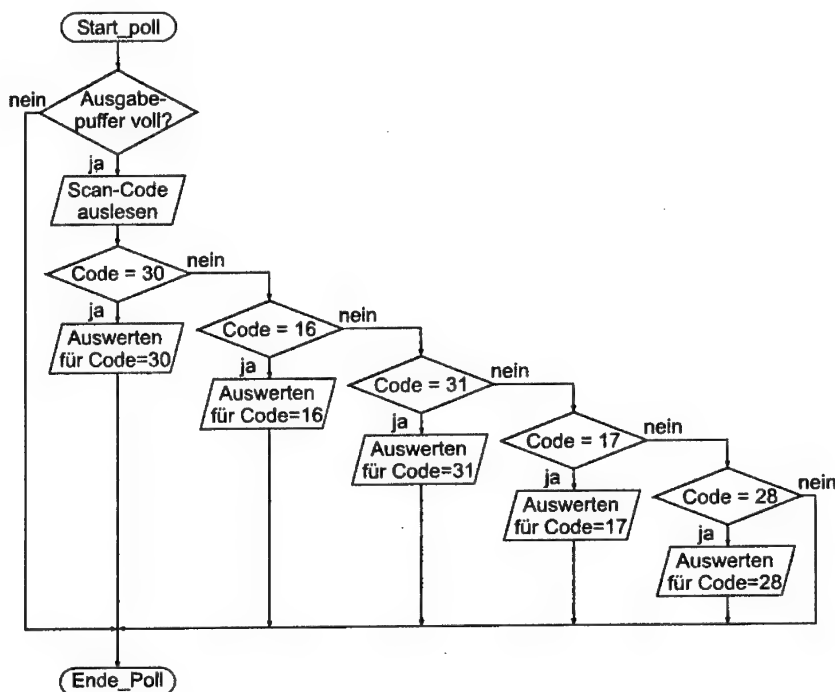


Bild 6.1: Programmflußplan für Tastatursteuerung zur Änderung von Sollwertamplitude und Ausgangsspannungsfrequenz

6.2 Multi-I/O-Karten

In Abschnitt 3.4.2 wurden die verwendeten Einsteckkarten und deren Funktionen (Zeitwerke, analoge und digitale Ein- und Ausgabe) bereits vorgestellt. Insbesondere bei Regelungen ist die Rückführung von Meßwerten in das Rechnersystem wichtig. Es ist auch durchaus denkbar, daß beispielsweise Sollwerte für Frequenz oder Spannungsamplitude über analoge Größen vorgegeben werden. Die Wandlung solcher Größen in digitale Werte wird zum

gleichen Zeitpunkt vorgenommen wie die der Ist-Werte einer Regelung, wie es in den Abschnitten 4.1 bzw. 4.3.4.2 und 5.3.3 beschrieben ist.

Solche Einsteckkarten bieten auch die Möglichkeit der Ein- bzw. Ausgabe von digitalen Daten. Die wichtigste Anwendung davon ist die Ausgabe des aktuellen Stromrichterzustandes, der zu den gewünschten Phasenumschaltungen führt. Da der Ein-/Ausgabebaustein aber mehrere unabhängige Ports enthält, können über ihn auch zusätzliche Daten eingelesen werden. Das können durchaus Signale sein, die den Zustand der Umgebung darstellen, wie z.B. ein Signal "Betriebsbereit" der Lastmaschine und ähnliches. Die über solche Ports eingelesenen Daten können jedoch keine Interrupts auslösen, sodaß auf diesem Weg keine zeitkritischen Daten übertragen werden sollten. Sie werden ebenfalls mit Hilfe von Pollingverfahren ausgelesen und weiterverarbeitet.

Schließlich enthält die Multi-I/O-Karte auch zwei Digital-Analog-Wandler zur Ausgabe von analogen Größen. Damit können beispielsweise analoge Sollwerte an die Steuerung der Lastmaschine vorgegeben werden.

6.3 Bildschirm

6.3.1 On-line-Ausgabe

Der Bildschirm ist das wichtigste Ausgabegerät von AT-Rechnern. Die Ausgaben erfolgen weitgehend über zeitaufwendige Softwareinterrupts, sodaß die Nutzung des Bildschirms während des zeitkritischen Stromrichterbetriebs im allgemeinen sehr eingeschränkt ist. Das gilt sowohl für den Betrieb des Bildschirms im Textmodus wie auch im Grafikmodus. Im unkritischen Off-line-Teil (z.B. während des Initialisierungsteils) der Software wird der Bildschirm dagegen zur interaktiven Dateneingabe genutzt, z.B. zur Auswahl des Pulsverfahrens oder der Sollwerte für Modulationsgrad bzw. Frequenz der Ausgangsspannung und Pulsfrequenz.

Heutige Graphikkarten verfügen über einen eigenen Speicherbereich, der über outp-Befehle angesprochen werden kann. Der Aufbau des Graphikspeichers und die Funktionsweise von Graphikkarten ist in der Literatur (z.B. [6.3]) ausführlich beschrieben. Im Textmodus stellt jede Speicherzelle (= 1 Byte) ein Zeichen nach dem ASCII-Standard auf dem Bildschirm dar. An jeder Stelle des Bildschirms kann also eines von 256 verschiedenen Zeichen ausgegeben und mit einem einzigen outp-Befehl verändert werden. Auf diese Weise ist es möglich ohne die Nutzung von vergleichsweise zeitaufwendigen Softwareinterrupts Bildschirmanzeigen zu

verwirklichen. Für die Anzeige von Zahlen (Integer- und Fließkommazahlen) müssen sie zuerst mit Hilfe geeigneter Funktionen in Zeichenketten umgewandelt werden. Die Programmiersprache C bietet dazu bereits standardmäßig entsprechende Routinen an. Bei Kenntnis des Zeitbedarfs der Ausgabe beispielsweise einer Fließkommazahl kann ein entsprechendes Zeitfenster gemäß Abschnitt 5.7.4 ermittelt und die Ausgabe darin ausgeführt werden.

Im Prinzip ist auch die Nutzung des Bildschirms im Graphikmodus denkbar. Hier stellt eine Speicherzelle nur ein Pixel dar. Zur Ausgabe von Linien muß deshalb beispielsweise eine große Anzahl an Speicherzellen beschrieben werden, was auch bei Verwendung des outp-Befehls relativ viel Zeit beansprucht¹. Die Nutzung des Bildschirms im Graphikmodus ist daher nur in Ausnahmefällen sinnvoll.

In dieser Arbeit wird der Bildschirm hauptsächlich für nicht-veränderliche Anzeigen wie z.B. der Bezeichnung des verwendeten Pulsverfahrens benutzt. Aufgrund der in Abschnitt 6.1 beschriebenen Möglichkeit der Änderung beispielsweise des Modulationsgrads, wurde auch eine On-line-Anzeige der Amplitude der Sollwertspannung, des Modulationsgrads, des Aussteuerungsgrads sowie der Ausgangsspannungsfrequenz realisiert. Um einen Scrolleffekt zu vermeiden, wurde immer derselbe Text ausgegeben, wobei aber auf einen Cursor-Rücklauf verzichtet wird. Dadurch bleibt der Cursor und damit der auszugebende Text immer in der gleichen Bildschirmzeile.

6.3.2 Off-line-Ausgabe

Es ist mit dem vorgestellten System auch möglich Daten während des Stromrichterbetriebs zu sammeln und sie dann nach dessen Beendigung off-line auszugeben. Das ist insofern interessant, als diese Möglichkeit zur Off-line-Datenüberwachung² und dadurch zur Fehlersuche verwendet werden kann.

Dazu wird eine Struktur definiert, welche die aufzuzeichnenden Variablen enthält, und ein entsprechend großer Speicherbereich dafür reserviert. Nun können z.B. in jedem Berechnungszeitraum die aufzuzeichnenden Werte den entsprechenden Elementen der Struktur zugewiesen werden. Bei solchen zyklischen Vorgängen ist es ausreichend den Speicherbereich

¹Die Ausgabe einer Horizontallinie benötigt 640 outp-Befehle bei VGA-Auflösung. Das entspricht einem Zeitbedarf von ca. 384 µs.

²Aufgrund der maskierten Hardware-Interrupts und des On-line-Betriebs ist on-line debuggen mit dem integrierten Debugger der Entwicklungsumgebung und damit auch die On-line-Überwachung von Variablen nicht möglich.

so groß zu wählen, daß etwa zwei bis drei Perioden der Ausgangsspannung überwacht werden können. Je nach Größe der Struktur kann der angeforderte Speicherbereich schnell die Grenzen des Compilers bzw. des Speichermodells erreichen. Es muß daher immer überprüft werden, ob die Reservierung auch tatsächlich durchgeführt wurde. Nach dem Abschalten des Stromrichters können die gespeicherten Variablenwerte zur weiteren Verarbeitung zusätzlich auf die Festplatte übertragen werden.

Mit der geeigneten Meßtechnik ist auch die Durchführung von Meßreihen mit dem vorgestellten Prinzip möglich. So können beispielsweise die Phasenströme über die Multi-I/O-Karte eingelesen und im reservierten Speicherbereich abgelegt werden. Die Messung kann über eine geeignete (Tastatur-)Steuerung zu einem beliebigen Zeitpunkt gestartet werden, sodaß ggf. auch Sonderereignisse berücksichtigt werden können. Nach dem Abschalten des Stromrichters kann über geeignete Routinen sogar eine graphische Auswertung beispielsweise in Form von Raumzeigerdarstellungen erfolgen.

6.4 Interrupts

Interrupts, wie sie in Abschnitt 2.3.2 beschrieben sind, bieten die schnellste und unmittelbarste Eingriffsmöglichkeit in das PC-System. Sie veranlassen das System nahezu sofort zur Auswertung der entsprechenden Information. Da dies ggf. den Betrieb des Stromrichters gefährden kann, sollte man diese Möglichkeit - abgesehen von den gewünschten Interrupts eines Zeitwerks - nur bedingt gebrauchen.

Die Einsprunzeit vom normalen Programmfluß in eine Interruptroutine liegt bei PentiumII-Systemen entsprechend Abschnitt 5.4.2 bei ca. 2 μ s, wenn der Interrupt über den entsprechenden Anschlußpunkt des ISA-Busses angefordert wird. Das entspricht etwa auch dem Zeitbereich der zulässigen Dauer eines Kurzschlusses der verwendeten Phasenschalter, sodaß diese Möglichkeit für den Aufbau eines direkt in das System eingreifenden Schutzmechanismus' entsprechend Abschnitt 7 genutzt werden kann.

6.5 Schnittstellen

Im allgemeinen sind die Schnittstellen des AT-Modells (serielle, parallele und USB-Schnittstelle) insbesondere durch die Anschlußmöglichkeit von Druckern ein wichtiges Ein- bzw. Ausgabegerät. Es können so beispielsweise Betriebsdaten auf unkomplizierte Weise dokumentiert werden.

Die einfache Handhabung der Schnittstellen des AT-Modells ist abhängig von der Möglichkeit vorhandene Funktionen des Betriebssystems, des BIOS oder der Programmiersprache verwenden zu können. Solche Funktionen werden meist von den Hardware-Interruptroutinen aufgerufen, welche die Schnittstellen bedienen. Sie berücksichtigen meist auch mögliche Fehler des Schnittstellenbetriebs, was die Leistungsfähigkeit der Schnittstelle beeinträchtigt. Ähnlich wie bei Tastatureingaben lassen sich Schnittstellen auch über Portadressen vollständig programmieren, sodaß kleine und vorher festgelegte Datenpakete ggf. schneller übertragen werden.

6.6 Netzwerkkarten

Im Prinzip gelten hier die gleichen Gesichtspunkte wie auch bei den vorherigen Ein-/Ausgaberäten bezüglich ihrer Echtzeittauglichkeit. In dieser Arbeit wurde diese Möglichkeit nicht behandelt und wird hier nur der Vollständigkeit halber genannt.

7 Schutzeinrichtungen

Schutzvorkehrungen sind die wichtigsten Einrichtungen, um Schäden an Anlage und nicht zuletzt auch am Menschen zu vermeiden. Wie bereits in Abschnitt 5 angesprochen, benötigt die Schutzeinrichtung - sofern sie in das Steuergerät "PC" einwirkt - vergleichsweise wenig Rechenzeit, muß aber um so höher priorisiert werden.

7.1 NOT-AUS

Diese Schutzeinrichtung schaltet die Schaltschränke der Last- und der Antriebsmaschine spannungsfrei, indem die Versorgungsleitung der Netzschränke jeweils unterbrochen wird. Dadurch wird auch die Entriegelung der mechanischen Bremse der Lastmaschine aufgehoben, was ein ungeführtes Austrudeln des Maschinensatzes im Notfall verhindert.

Da Schütze im allgemeinen langsam, d.h. im Millisekundenbereich, schalten, kann dies kein wirkungsvoller Schutz für Halbleiterbauelemente darstellen, deren Kurzschlußfestigkeit nur für sehr kurze Zeit (1-2 μ s) gewährleistet ist. Die NOT-AUS-Einrichtung soll den Maschinensatz, vor allem aber den Bediener vor den Auswirkungen unvorhergesehener Fehler und Schäden schützen.

7.2 Halbleiterschutz

7.2.1 Direkter Schutz mit Interrupt

Zu hohe Ströme führen schnell zu Schäden bei elektronischen Schaltern, da Halbleiterbauelemente nur bedingt kurzschlußfest sind. Sie können meist nur sehr begrenzte Zeit einen vergleichsweise hohen Kurzschlußstrom führen und müssen deshalb innerhalb einiger 100 ns bzw. 1-2 μ s abgeschaltet werden. Wie oben erwähnt, ist das mit mechanischen Schaltern nicht realisierbar.

Im vorliegenden Aufbau ist der Halbleiterschutz bezüglich Überströmen bereits in die IGBT-Ansteuermodule integriert. Dort wird die Kollektor-Emitter-Spannung u_{CE} der einzelnen IGBT überwacht [7.1]. Führt beispielsweise ein Fehler der Anlage durch Überströme zu einem überhöhten Wert von u_{CE} , dann werden beide IGBT eines Phasenschalters ausgeschaltet und der Anschlußpunkt "Error" (Fehlensignal) des Ansteuermoduls dann auf 0V geschaltet. Die

Fehlersignale aller drei Module wurden über einen NAND-Logikbaustein zu einem Signal zusammengefaßt, das am PC direkt einen Interrupt auslöst. Interrupts unterbrechen sofort den aktuellen Programmteil und führen die zugeordnete Interruptroutine aus. Entsprechend Bild 7.1 gibt sie sofort den Stromrichterzustand "00 00 00" (alle Transistoren AUS) aus, sodaß es zu keinen unbeabsichtigten Einschaltungen im Fehlerfall mehr kommen kann und veranlaßt das Beenden des Hauptprogramms. Da die Einsprunzeit in eine Interruptroutine ca. 2 µs beträgt, kann diese Vorgehensweise sehr gut angewandt werden.

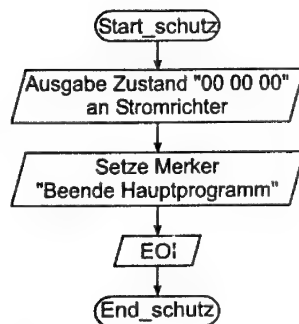


Bild 7.1:
Programmflußplan für die Interruptroutine "Schutz"

Die Programmierung der Interruptroutine erfolgt nach Anhang B und wird ähnlich wie die Interruptroutine zur Behandlung von Steuereingriffen vor dem eigentlichen Stromrichterbetrieb initialisiert. Auch sie wird ausschließlich nur dann aufgerufen, wenn ein entsprechender "Weckruf" sie aktiviert. In diesem Fall handelt es sich dabei um ein zeitlich nicht vorhersagbares Ereignis, weshalb sich die Interrupttechnik auch in dieser Hinsicht hier besonders gut eignet.

7.2.2 Indirekter Schutz durch Softwareabschaltung

Die Phasenströme werden für die Regelung in festen Abständen von der Multi-I/O-Karte gemessen. Auf diese Weise kann die Software überprüfen, ob ggf. zu hohe Ströme fließen und daher die IGBT abgeschaltet werden müssen. Die Messungen werden meist an den Halbperiodengrenzen durchgeführt, sodaß eine maximale Totzeit in der Höhe der Halbperiodendauer verstreichen kann ehe ein unzulässiger Betriebspunkt erfaßt wird. Deshalb ist diese Methode zur Schnellabschaltung von elektronischen Leistungsschaltern nicht anwendbar. Sie eignet sich aber durchaus, um länger dauernden Überlastbetrieb zu erkennen und ggf. Gegenmaßnahmen zu ergreifen.

8 Experimentelle Untersuchungen

Alle in diesem Abschnitt gezeigten Messungen wurden im gesteuerten Betrieb unter Vorgabe eines sinusförmigen Verlaufs der Einzelspannungen durchgeführt. Dies ist auch als ein gleichmäßig auf einer Kreiskurve umlaufender Spannungsraumzeiger darstellbar. Die angegebenen Kurvenverläufe werden jeweils durch das verwendete Pulsverfahren, den Modulationsgrad M und den Winkel α , den der Spannungsraumzeiger mit der Realteil-Achse einschließt, charakterisiert. Nach Abschnitt 4.3.5 darf der Modulationsgrad M nicht mit dem Modulationsgrad m einer Phase verwechselt werden. Während M im stationären Betrieb konstant ist und immer Werte zwischen 0 und $4/\pi$ einnimmt, ist m jeweils nur für einen Zeitpunkt bzw. für eine Pulsperiode oder Halbperiode definiert und stets ein Wert zwischen -1 und 1. Sofern die Frequenz f der Ausgangsspannung ebenfalls eine wichtige Beschreibungsgröße darstellt, ist auch sie mit angegeben. Alle Messungen wurden mit belasteter Maschine durchgeführt, soweit nichts anderes vermerkt ist.

8.1 Meßaufbau und Meßtechnik

Bild 8.1 zeigt die Anordnung des Laboraufbaus mit Wechselrichter und Kurzschlußläufermaschine. Beides ist bereits in Abschnitt 3 näher beschrieben. Im Bild nicht enthalten ist die Lastmaschine, die mit einem externen Gerät gesteuert wird und ebenfalls in Abschnitt 3 vorgestellt wurde.

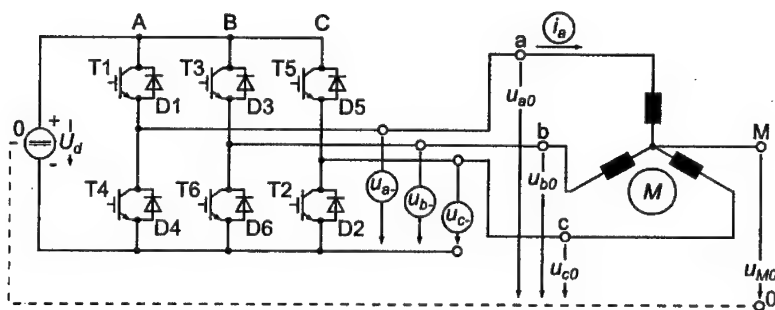


Bild 8.1: Anordnung von Wechselrichter und Asynchronmaschine mit den gemessenen Spannungen u_a , u_b , u_c und des Stroms i_a .

Die hier gezeigten Messungen wurden mit Meßgeräten erzielt, die nicht fester Bestandteil des Antriebssystems sind¹. Direkt gemessen wurden die eingekreisten Größen, nämlich die Phasenpotentiale u_a , u_b , u_c und der Stroms i_a . Die Phasenpotentiale wurden in den folgenden Darstellungen immer gegen den negativen Pol der Zwischenkreisspannung U_d gemessen. Dadurch ergibt sich ein fester Bezugspunkt, sodaß Verschiebungen des Bezugspotentials und damit Meßfehler klein gehalten werden. Im Gegensatz zu den Mittelpunktspannungen u_{aM} , u_{bM} , u_{cM} enthalten sie neben der durch die verschiedenen Pulsverfahren verursachte Nullkomponente zusätzlich noch eine Gleichspannung $U_d/2$. In die folgenden Bilder wurden auch die Koordinaten für die Phasenpotentiale u_{a0} , u_{b0} , u_{c0} gegenüber dem fiktiven Nullpunkt der Zwischenkreisspannung eingetragen, die nur die durch die Pulsverfahren hervorgerufene Nullkomponente enthalten. Die Mittelpunktspannung u_{M0} entspricht dabei genau der Nullkomponente. Die Spannungen u_{aM} , u_{bM} , u_{cM} und die verketteten Spannungen u_{ab} , u_{bc} , u_{ca} wurden bei den Untersuchungen ebenfalls gemessen. Da diese Größen von allen drei Umschaltvorgängen beeinflusst werden, ist ihre Analyse wesentlich komplexer als die der gezeigten Größen. Deshalb werden sie in den folgenden Bildern nicht dargestellt.

8.1.1 Meßgeräte

Das zentrale Meßgerät ist ein digitales Speicher-Oszilloskop HP 54512B mit vier Meßeingängen. Es verfügt über eine Abtastrate von 10^9 s^{-1} . Damit sind auch Vorgänge in Mikrosekundenbereich sehr gut darstellbar. Durch den zum Teil erheblichen Unterschied der Periodendauer der Ausgangsspannung (z.B. $16 \frac{2}{3} \text{ ms}$ für 60 Hz) zur Pulsfrequenz (z.B. $200 \mu\text{s}$ bei 5 kHz) ergibt sich jedoch ein Konflikt, da beides nicht gleichzeitig mit ausreichender Auflösung dargestellt werden kann.

Im sogenannten "Real-Time"-Modus des oben genannten Oszilloskops [8.1] werden für jeden Kanal 8000 Meßwerte in der eingegebenen Zeitablenkung aufgenommen, während es im "Repetitive"-Modus lediglich 500 sind. Davon werden in beiden Modi in der Anzeige des Oszilloskops jedoch nur 500 Werte angezeigt. Im "Real-Time"-Modus können die übrigen 7500 Werte über die Änderung der "Delay"-Einstellung angezeigt werden. Wird beispielsweise eine 60 Hz-Schwingung bei einem Anzeigezeitraum des Oszilloskops von 20 ms gemessen, so werden insgesamt $8000 / (16.67/20 \cdot 500) = 19.2$ Schwingungen aufgenommen. Mit dieser Einstellung erfolgt alle $40 \mu\text{s}$ eine Abtastung. Dadurch können Pulse, die kürzer als $40 \mu\text{s}$ sind, nicht mehr sicher erfaßt werden, sodaß sich beispielsweise im Bereich hoher Modulationsgrade die

¹Feste Bestandteile des Antriebs sind diejenigen Sensoren, deren Ausgangswerte für die Steuerung und Regelung notwendig sind wie beispielsweise LEM-Module zur Messung der drei Phasenströme.

gepulsten Spannungsverläufe nur unvollständig darstellen lassen. Es ist daher im vorliegenden Fall bei Verwendung des Real-Time-Modus sinnvoller, eine kleinere Zeitablenkung zu wählen, die so bemessen ist, daß die 8000 Abtastwerte zwischen einer und zwei 60 Hz-Schwingungen darstellen. Das führt hier schließlich auf einen Anzeigzeitraum von 2 ms, sodaß 8000 Abtastpunkte einen Gesamtzeitraum von 32 ms repräsentieren. Das bedeutet, daß alle $32 \text{ ms} / 8000 = 4 \text{ } \mu\text{s}$ eine Abtastung erfolgt. Dadurch werden lediglich Pulse einer Dauer von unter $4 \text{ } \mu\text{s}$ nicht mehr sicher erfaßt, was eine erhebliche Verbesserung gegenüber des Repetitive-Modus darstellt.

Zur Darstellung von Vorgängen, die im Bereich der Pulsperiodendauer liegen, reicht die Darstellung im Repetitive-Modus aus, da hier nur Aussagen über den Verlauf der Phasenspannungen innerhalb eines vergleichsweise kleinen Zeitbereichs getroffen werden und die Periodendauer der Ausgangsspannung daher nicht von Interesse ist.

Die im Oszilloskop gespeicherten Daten werden mittels eines zweiten PC über den sogenannten GPIB-Bus ausgelesen und abgespeichert, sodaß sie zur weiteren Auswertung zur Verfügung stehen. Der oben beschriebene Vorteil des Real-Time-Modus geht aber mit dem Nachteil einher, daß sehr viel größere Dateien erzeugt werden (160 kB) als im Repetitive-Modus (10 kB). Dies erfordert leistungsfähige Rechner um solche Datenmengen effizient handhaben zu können.

8.1.2 Spannungsmessung

Die einzelnen Spannungen wurden mit Trenn-Tastköpfen vom Typ SI-9000, bei denen Maximalwerte am Eingang von 1000 V Gleichspannung bzw. 700 V Wechselspannung zulässig sind, gemessen. Sie besitzen eine Bandbreite von 25 MHz und einen umschaltbaren Spannungsteiler von 1:10 auf 1:100. Damit können alle Spannungen am Wechselrichter und an der Maschine direkt, d.h. ohne zusätzlichen externen Spannungsteiler am Oszilloskop gemessen werden.

8.1.3 Strommessung

Bei der Strommessung ist ebenfalls eine Potentialtrennung erforderlich. Dies wurde mit Stromzangen vom Typ A6303 der Firma Tektronix realisiert. Zum Betrieb der Stromzange ist ein zusätzlicher Verstärker vom Typ AM503 (Fa. Tektronix) erforderlich, der das Ausgangssignal der Stromzange für die Anzeige in Oszilloskopen entsprechend aufbereitet. Die Bandbreite der Kombination aus Zange und Verstärker beträgt laut Datenblätter ([8.3],[8.4]) 15 MHz. Dies reicht aus, um den Oberschwingungsanteil, der durch die Pulsverfahren ent-

steht, gut darstellen zu können, zumal durch die Abtastung im Oszilloskop gemäß Abschnitt 8.1.1 zusätzliche Fehler entstehen.

8.2 Messungen zur Halbperiodensteuerung

Bei der Halbperiodensteuerung kann der Wechsel von 60° -Sektoren, die durch zwei benachbarte Spannungsraumzeiger aufgespannt werden (siehe Bild 4.6), im Prinzip nach beiden Typen von Halbperioden stattfinden. Das kann am einfachsten bei zweiphasigen Pulsverfahren beobachtet werden, da die Phase, die nicht gepulst wird, ebenfalls dort wechselt. Die Teilbilder in der linken und rechten Spalte von Bild 8.2 zeigen die Spannungen u_a , u_b , u_c , die sich durch zweiphasiges Pulsen erster Art ergeben. Für zweiphasiges Pulsen zweiter Art ergeben sich ähnliche Verläufe. In beiden Spalten wird ein Spannungsraumzeiger umgesetzt, der sich in der Nähe des Winkels $\alpha = 240^\circ$ mit der Realteil-Achse befindet. Auf der rechten Werteachse ist jeweils die Phasenspannung bezogen auf den fiktiven Nullpunkt der Zwischenkreisspannung angetragen. Der Anschaulichkeit halber wurden außerdem die drei Kurven so dargestellt, daß die Gitterlinien in guter Näherung auf die Halbperiodengrenzen fallen, wobei der Typ der Halbperioden in der untersten Zeile des Diagramms der Phase c eingetragen ist.

Die linken Verläufe in Bild 8.2 zeigen den Sektorwechsel zum Zeitpunkt $t = 0.5$ ms nach einer Halbperiode vom Typ A bei zweiphasigem Pulsen erster Art. Wie zu sehen ist, wird in Phase a dort nur noch der Teil des Pulses erzeugt, der zum Sektor mit der Nummer 4 gehört. Die darauf folgende Halbperiode bildet bereits einen Spannungsraumzeiger ab, der zum Sektor mit der Nummer 5 gehört. In diesem Bereich wird die Phase a jedoch nicht mehr gepulst.

Die Bilder der rechten Spalte zeigen die Spannungsverläufe bei gleichen Bedingungen, jedoch findet hier der Wechsel der Sektoren nach einer Halbperiode vom Typ B statt. Während die Phase c in Zeiten $t < 0.5$ ms noch voll angesteuert wird, wechselt der Sektor zum Zeitpunkt $t = 0.5$ ms. Es ist zu erkennen, daß im Anschluß daran eine Halbperiode vom Typ A folgt, in der die Phase wieder geschaltet wird.

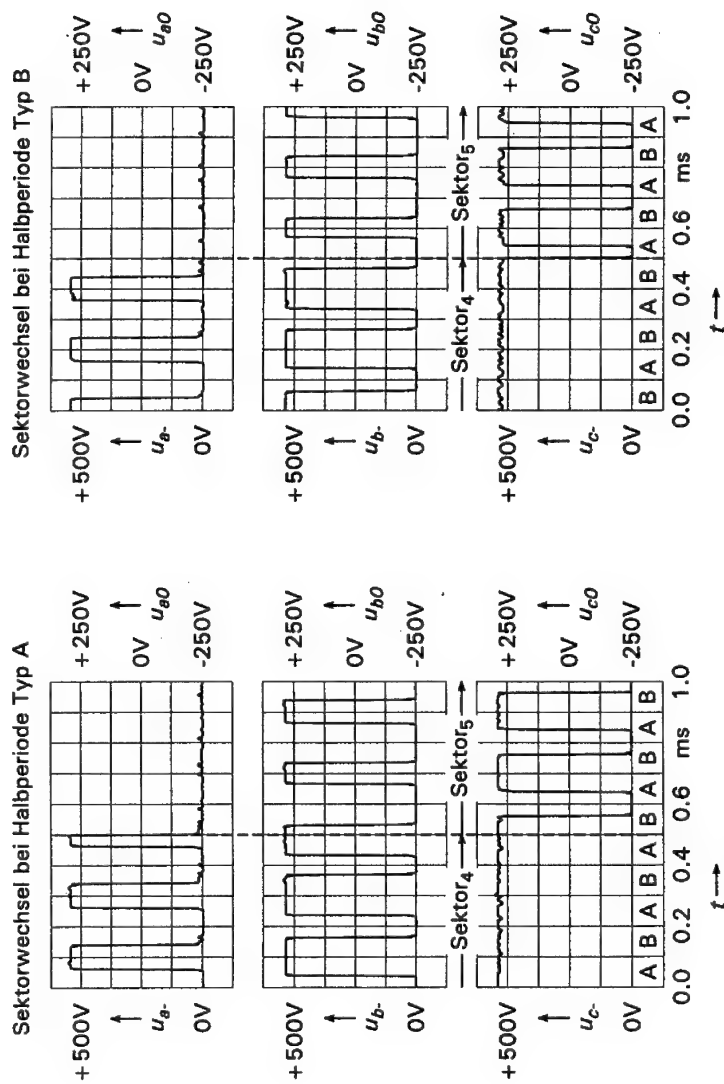


Bild 8.2: Darstellung der Sektorgrenze bei zweiphasigem Pulsen erster Art ($M = 0.7$, $\alpha = 240^\circ$):
 links: Wechsel am Ende des Sektors 4 nach Halbperiode vom Typ A
 rechts: Wechsel am Ende des Sektors 4 nach Halbperiode vom Typ B

8.3 Messungen zu den Sonderfällen der Stromrichtersteuerung

8.3.1 Warteschleife

Abschnitt 5.5.2.2 geht ausführlich auf die Funktion und Notwendigkeit von Warteschleifen ein. Bild 8.3 zeigt den Verlauf der Spannungen u_a , u_b und u_c sowie das Ausgangssignal u_{Timer} des Zeitwerks, welches bei Erreichen des HIGH-Zustandes den Interrupt im PC nach Abschnitt 2.3.2 auslöst, für einen Fall, in dem mehrere Warteschleifen auftreten. Die Verläufe der Phasenspannungen und des Ausgangssignals des Zeitwerks wurden wieder so verschoben, daß die Halbperiodengrenzen mit dem Gitter des Diagramms gut übereinstimmen.

Kurz vor dem Ende der Halbperiode¹ leitet das Zeitwerk zum Zeitpunkt $t \approx 37 \mu s$ einen Steuereingriff ein, der die Umschaltung der Phase b bewirkt. Die Überprüfung des nächsten zu ladenden Steuereingriffs-Intervalls ergibt, daß eine Verzögerung zweiter Art nach Abschnitt 5.5.2.2 vorliegt und daher der nächste Steuereingriff nicht mehr zeitpunktgenau ausgeführt werden kann. Das Zeitwerk wird also nicht geladen

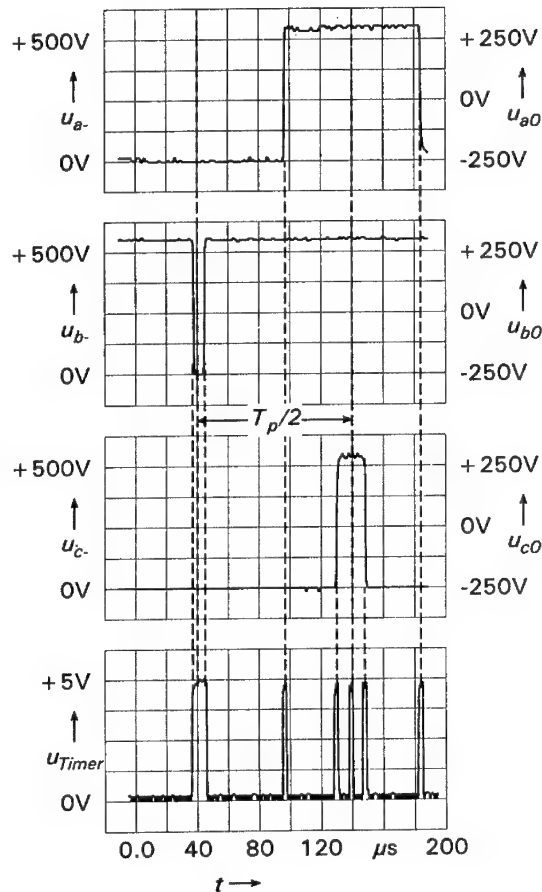


Bild 8.3: Warteschleife mit drei Steuereingriffen ($t = 40 \mu s$) bei Sinusmodulation der Einzelphasen ($\alpha = 90^\circ$, $M = 0.97$)

¹Der Beginn einer Halbperiode ist bei Sinusmodulation der Einzelphasen immer im Bereich der Normalzustände nach Abschnitt 5.5.3

und somit bleibt auch dessen Ausgangssignal auf 5V. Statt dessen wird eine Warteschleife ausgeführt, an deren Ende die Bearbeitung des zweiten Steuereingriffs¹ steht. Die erneute Überprüfung der Bedingung für Verzögerungen zweiter Art des nächsten Steuereingriff-Intervalls führt wiederum auf eine Warteschleife in deren Folge der dritte Steuereingriff, der wiederum die Phase b schaltet, ausgeführt wird. Erst das Steuereingriff-Intervall, das sich von $t \approx 45 \mu\text{s}$ bis $t \approx 96 \mu\text{s}$ erstreckt, benötigt keine Warteschleife mehr, sodaß dieses Intervall wieder in das Zeitwerk geladen wird und das zugehörige Ausgangssignal u_{Timer} wieder zurück auf 0V springt.

Vor Ende der Halbperiode, d.h. zum Zeitpunkt $t \approx 130 \mu\text{s}$ führt ein Interrupt zur Umschaltung der Phase c. Obwohl das nächste Steuereingriff-Intervall mit ca. $10 \mu\text{s}$ vergleichsweise kurz ist, liegt keine Verzögerung zweiter Art vor, d.h. der folgende Steuereingriff kann rechtzeitig zum Zeitpunkt $t \approx 140 \mu\text{s}$ ausgeführt werden. Eine Warteschleife ist also überflüssig. Dieselben Bedingungen gelten ebenfalls für den darauffolgenden Steuereingriff bei $t \approx 149 \mu\text{s}$.

Die vorgegebene Halbperiodendauer $T_p/2 = 100 \mu\text{s}$ wird exakt eingehalten. Das zeigt, daß die in Abschnitt 5.4.2 beschriebene Analyse der Interrupts und deren Zeitbedarf richtig interpretiert wurde und die in Abschnitt 5.5.2.2 vorgeschlagene Lösung zur Vermeidung von Verzögerungen zweiter Art mittels Warteschleifen zu einem guten Ergebnis führt.

8.3.2 Umschaltunterdrückung

Umschaltunterdrückungen - wie in Abschnitt 5.5.3 beschrieben - sind eine wichtige Technik zur Realisierung von zwei- und einphasigen Pulsverfahren. Aber auch bei Sinusmodulation der Einzelphasen ist es beispielsweise bei passiver Begrenzung nach Abschnitt 5.7.5 möglich, daß zeitweise Bereiche auftreten, in denen bei diesem Pulsverfahren der Betrag des Modulationsgrads $|m|$ einer Phase größer oder gleich 1 ist und demzufolge der entsprechende Umschaltzeitpunkt genau auf eine der Grenzen oder sogar in Bereiche außerhalb der Halbperiode fallen würde. Das kann sowohl bei Halbperioden vom Typ A wie auch vom Typ B der Fall sein. Solche Umschaltzeitpunkte können aber innerhalb des jeweiligen Ausführungszeitraums nicht realisiert werden und müssen daher unterdrückt werden. Dazu wurde in Abschnitt 5.5.3 der sogenannte Normalzustand z_{norm} und der ggf. davon abweichende Anfangszustand z_{Anfang} definiert und eingeführt.

¹In diesem Fall ist der nächste Steuereingriff vom Typ Halbperiodenende. Die Steuereingriff-Typen spielen aber für die Warteschleife selbst keine Rolle

Bild 8.4 zeigt einen Fall von passiver Begrenzung bei Sinusmodulation der Einzelphasen. Die beiden Teilbilder stellen einen Ausschnitt der Phasenspannung u_a bei $M = 1.01$ und $\alpha \approx 270^\circ$ entsprechend Bild 8.6 dar. Bei $t \approx 0$ ms, $t \approx 0.2$ ms und $t \approx 0.4$ ms bilden sich im oberen Verlauf noch schmalere werdende Spannungspulse aus, während die nächsten Pulse bei $t \approx 0.6$ ms, $t \approx 0.8$ ms und $t \approx 1.0$ ms bereits unterdrückt werden. Der Verlauf im unteren Diagramm von Bild 8.4 zeigt den Zeitpunkt des Wiederbeginns des Pulsens der Phase a bei $t \approx 0.5$ ms. Zu Zeiten $t < 0.5$ ms könnten auch hier Spannungspulse erscheinen, die jedoch aufgrund der Umschaltunterdrückung entfallen. Aufgrund der Halbperiodensteuerung ist es durchaus möglich, daß der letzte Spannungspuls bei $t \approx 0.4$ ms nur Teil der Halbperiode von $t \approx 0.3$ ms bis $t \approx 0.4$ ms ist und der Teil des Pulses, der zur darauf folgenden Halbperiode gehört, bereits ebenfalls unterdrückt wird. Aus dem oberen Bild in 8.4 ist dies jedoch nicht eindeutig erkennbar.

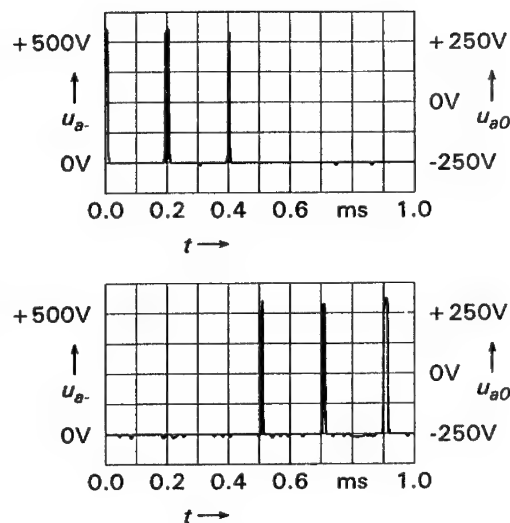


Bild 8.4: Beginn (oben) und Ende (unten) der Umschaltunterdrückung bei Sinusmodulation der Einzelphasen ($\alpha \approx 270^\circ$, $M = 1.01$)

Umschaltunterdrückungen, die zu Spannungspulsen führen, die nur Teil einer Halbperiode sind, können aber sehr einfach bei zweiphasigen Pulsverfahren nachgewiesen werden. Bild 8.5 zeigt die Auswirkung der Umschaltunterdrückung für Halbperioden des Typs A (Bild unten) und des Typs B (Bild oben) bei zweiphasigem Pulsen erster Art. Die Spannungsverläufe wurden wieder so auf der Zeitachse verschoben, daß die Halbperiodengrenzen in guter Näherung mit den Gitterlinien übereinstimmen.

Das obere Bild in 8.5 zeigt den Wechsel vom Pulsbetrieb zu einem Bereich mit Modulationsgraden $m \leq -1$. Der Übergang findet genau an einer Halbperiodengrenze statt ($t = 0.5$ ms), welche der Beginn einer Halbperiode vom Typ B ist. Es wird nur der Teil des Pulses ausgeführt, der zur vorherigen Halbperiode gehört ($t = 0.4$ ms bis $t = 0.5$ ms). Für die darauf folgende Halbperiode wird aber erkannt, daß $m \leq -1$ ist und deshalb der Umschaltzeitpunkt außerhalb des Ausführungszeitraums liegt, weshalb die zugehörige Umschaltung unterdrückt wer-

den muß. Dadurch beginnt der Verlauf der Phasenspannung in dieser Halbperiode nicht mit dem Normalzustand¹ $z_{\text{Anfang}} = z_{\text{normA}} = 10$, sondern mit dem Anfangszustand $z_{\text{Anfang}} = 01$. Es entsteht dadurch ein Spannungspuls, der am Ende der Halbperiode abbricht.

Das untere Bild in 8.5 zeigt einen ähnlichen Fall für Halbperioden vom Typ A. Hier ist wieder der Übergang vom gepulsten Betrieb in einen Bereich mit Modulationsgraden $m \leq -1$ dargestellt. Er findet aber im Gegensatz zum oberen Bild zu Beginn einer Halbperiode vom Typ A statt. In

diesem Fall stimmen der Anfangszustand und der Normalzustand bereits überein, sodaß sich hier kein Puls ausbildet, der nur zu einer Halbperiode gehört.

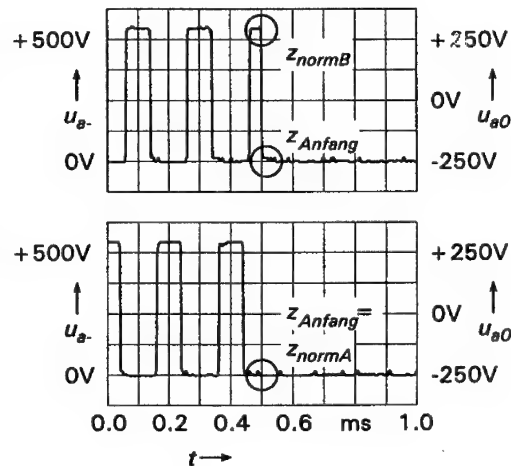


Bild 8.5: Umschaltunterdrückung zu Beginn einer Halbperiode vom Typ A (unten) und B (oben) bei zweiphasigem Pulsen erster Art ($\alpha \approx 240^\circ$, $M = 0.7$)

8.4 Betrieb des Asynchronmotors

Zunächst wird in diesem Abschnitt der Betrieb bei sinusförmigem Verlauf des Spannungssollwert-Raumzeigers diskutiert. Das bedeutet, daß der Modulationsgrad M bei Sinusmodulation der Einzelphasen den Wert 1 nicht überschreitet. Bei Verfahren mit Nullkomponente ist er entsprechend $2/\sqrt{3}$. Anschließend wird der Betrieb bei aktiver und passiver Begrenzung des Spannungssollwert-Raumzeigers entsprechend Abschnitt 5.7.5 dargestellt. Alle Meßreihen wurden für die vier in dieser Arbeit vorgestellten Pulsverfahren durchgeführt, sodaß deren Vergleich sehr leicht fällt. Die Reihenfolge der gezeigten Diagramme ist stets Sinusmodulation der Einzelphasen und Modulation mit symmetrischen Nullzuständen in der oberen Reihe sowie zweiphasiges Pulsen erster Art und zweiphasiges Pulsen zweiter Art in der unteren Reihe. Als Referenz der Spannungsverläufe dient Bild 8.6, das die jeweiligen Verläufe des zeit-

¹Nach Abschnitt 5.5.3 ist der Normalzustand einer Halbperiode auf den Stromrichter bezogen und daher in sechsstelliger Notation angegeben. In Anlehnung daran kann man aber auch einen zweistelligen Normalzustand einer einzelnen Phase definieren.

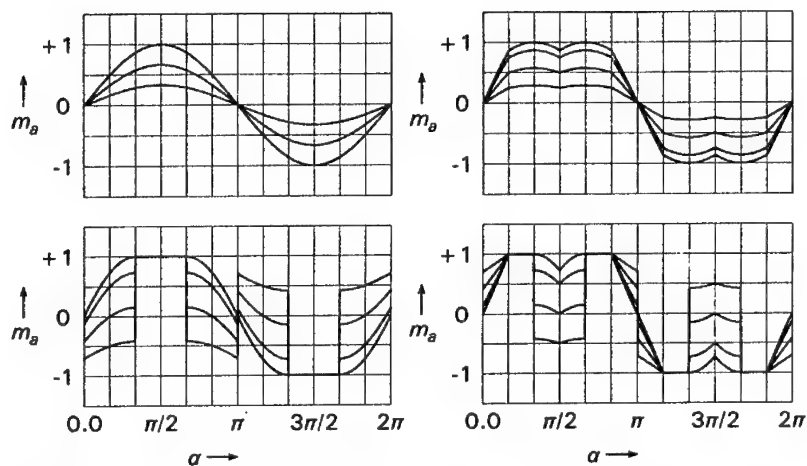


Bild 8.6: Verlauf des Modulationsgrads m für die vier Pulsverfahren (von oben links nach rechts unten: Sinusmodulation der Einzelphasen, Modulation mit symmetrischen Nullzuständen, zweiphasiges Pulsen erster Art, zweiphasiges Pulsen zweiter Art)

abhängigen Modulationsgrads m der Einzelphasen zeigt, wie sie auch in der Literatur (z.B. [8.5], [8.6]) dargestellt sind. Die Kurvenscharen zeigen dabei die Verläufe unter Variation des Betrags des Spannungsraumzeigers, der die Werte 0.33, 0.66, 1 und $2/\sqrt{3}$ annimmt.

Die Asynchronmaschine wurde im Test generell bei 60 Hz betrieben. Dies stellt bei einer Pulsfrequenz von 5 kHz einen guten Kompromiß zwischen der Forderung nach größtmöglicher Auflösung der Spannungspulse und der Messung ganzer Ausgangsschwingungen dar. Die Periodendauer beträgt $16\frac{2}{3}$ ms, sodaß im

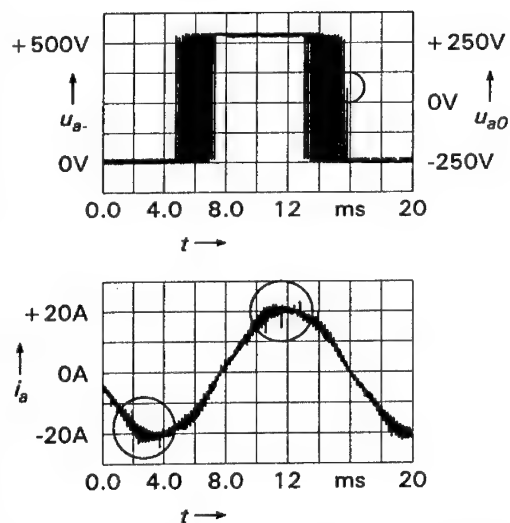


Bild 8.7: Beispiel einer Abtastung von Spannungspulsen an deren Flanke (oben) und einer stochastischen Erfassung von Stromspitzen (unten)

Real-Time-Modus eine Auflösung von $4\text{ }\mu\text{s}$ pro Abtastung erreicht werden kann. Trotzdem kann es auftreten, daß sehr kurze Spannungspulse nur an der Flanke oder sogar überhaupt nicht abgetastet werden. Das ist oben in Bild 8.7 an der eingekreisten Stelle der Fall. Ebenso existieren bei jeder Phasenumschaltung Stromspitzen von etwa $0.5\text{ }\mu\text{s}$ Dauer, wie sie in den eingekreisten Stellen unten in Bild 8.7 sichtbar werden. Bei der eingestellten Abtastrate des Oszilloskops werden sie aber nur selten erfaßt und treten je nach Abtastaugenblick des Oszillographen in den Bildern stochastisch auf. Diese Stromspitzen wurden nicht im einzelnen untersucht, es wurde aber bei Messungen mit höherer zeitlicher Auflösung festgestellt, daß sie bei jeder Phasenumschaltung auftreten.

8.4.1 Betrieb mit sinusförmigem Verlauf des Spannungs-Sollwert-Raumzeigers

Bild 8.8 zeigt die Spannungs- und Stromkurven der vier Pulsverfahren mit einem Modulationsgrad M , der knapp unterhalb des jeweils maximal möglichen Modulationsgrads liegt. Für die Sinusmodulation der Einzelphasen ist dies $M = 0.95$ und für die drei Pulsverfahren mit Nullkomponente $M = 1.1$. Mit diesen Modulationsgraden entstehen Spannungspulse, die wegen ihrer Dauer immer erfaßt werden. Wegen der Bildbreite wurde auf die Angabe einer zweiten Werte-Achse für die Spannung u_{a0} , wie sie in den Bildern 8.2 bis 8.5 eingezeichnet ist, verzichtet.

In der oberen Reihe sind die Spannungs- und Stromverläufe dargestellt, die durch dreiphasige Pulsverfahren hervorgerufen werden (Sinusmodulation der Einzelphasen links und Modulation mit symmetrischen Nullzuständen rechts). Deutlich zu erkennen sind in der unteren Reihe diejenigen Bereiche, in denen die Phase a nicht geschaltet wird. Bei zweiphasigem Pulsen erster Art (links) sind das in jeder Periode der Ausgangsspannung zwei Bereiche zu je 60° , während es bei zweiphasigem Pulsen zweiter Art jeweils vier Bereiche mit 30° sind. Es ist außerdem zu sehen, daß die Stromverläufe in der rechten Spalte tendenziell einen niedrigeren Oberschwingungsanteil enthalten als die der linken Spalte.

8.4.2 Betrieb bei kleinen Modulationsgraden ($M \approx 0.3$)

Diese Betriebsart tritt in der Praxis eigentlich nur bei kleinen Ausgangsfrequenzen auf. Mit der verwendeten Asynchronmaschine ist ein solcher Betrieb bei 60 Hz nur im Leerlauf möglich, da sich bereits hier ein Betriebspunkt in der Nähe des Kipp-Punktes einstellt. Eine Belastung der Maschine führt daher sehr schnell zum Abbremsen der Maschine bis zum Stillstand. Der Betrieb mit niedrigeren Frequenzen hätte zwar einerseits zur Folge, daß die Maschine belastet werden könnte, andererseits wäre die Periodendauer entsprechend größer, sodaß die Spannungspulse noch schlechter erkennbar und nur mit zum Teil sehr hohen stochastischen

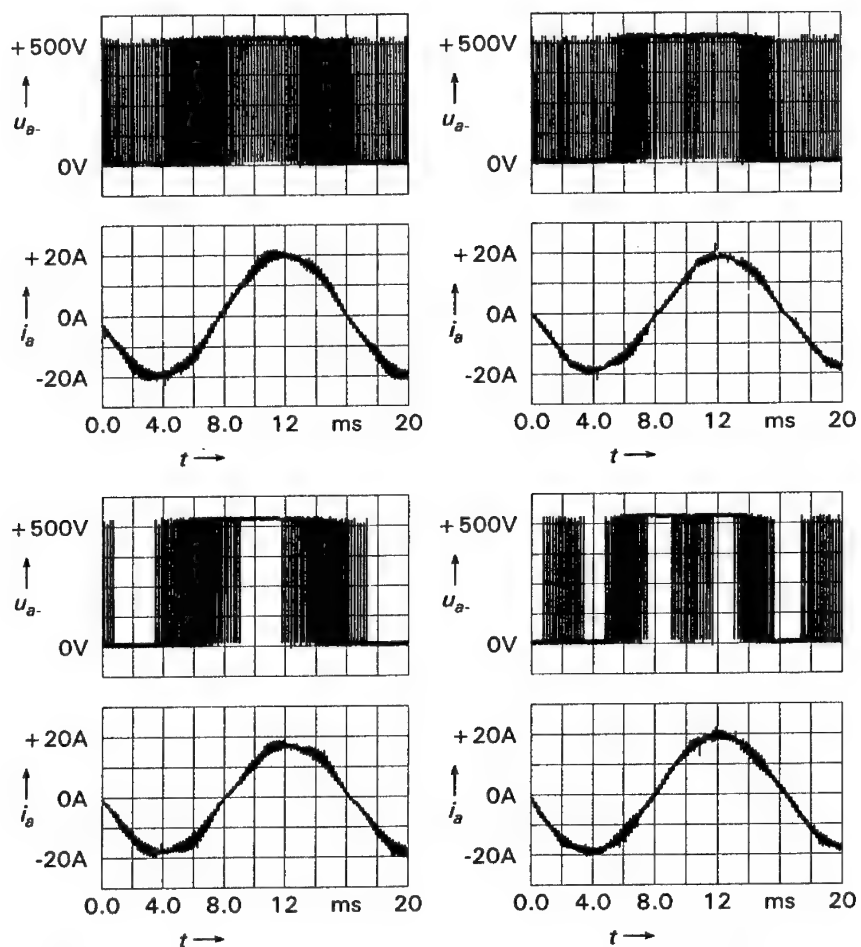


Bild 8.8: Verlauf der Spannung u_a und des Stroms i_a bei Modulationsgrad $M = 0.95$ (Sinusmodulation) bzw. $M = 1.1$ (Verfahren mit Nullkomponente) und einer Frequenz der Grundschiwingung von 60 Hz

Fehlern abgebildet werden könnten. Bild 8.9 zeigt die Spannungs- und Stromverläufe der einzelnen Pulsverfahren für diese Betriebsart. Die Verläufe der Phasenspannung u_a in der oberen Reihe kann aufgrund der Dauer der Spannungspulse und der Zeichengenauigkeit nicht mehr ausreichend aufgelöst werden, sodaß sich ein nahezu schwarzes Band ergibt. In der unteren Reihe ergeben sich durch die zweiphasigen Pulsverfahren schmalere Pulse, die immer noch gut aufgelöst werden können. Allerdings tritt hier besonders bei zweiphasigem Pulsen zweiter Art (unten rechts) die oben beschriebene Abtastungenauigkeit in der Phasenspannung u_a auf.

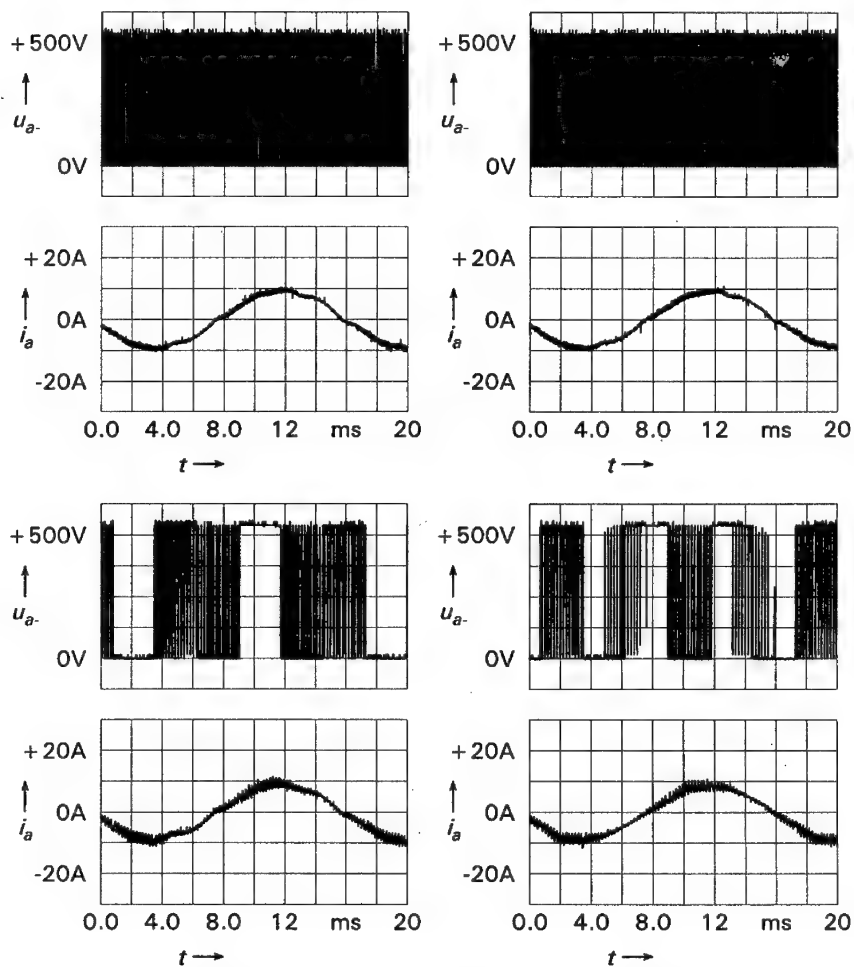


Bild 8.9: Verlauf der Spannung u_a und des Stroms i_a bei Modulationsgrad $M = 0.3$ und einer Frequenz der Grundschwingung von 60 Hz im Leerlauf

Da im Leerlauf das Verhältnis des Anteils der Stromgrundschwingung zum Anteil der Oberschwingungsströme vergleichsweise klein ist, treten die Oberschwingungsströme mehr in Erscheinung. Wie in der Literatur beschrieben, entstehen sie nicht nur durch den Pulsbetrieb (z.B. [8.7]) sondern auch durch die Maschine selbst (z.B. [8.8]). Dennoch sind sie bei zweiphasigem Pulsen zweiter Art (unten rechts) vergleichsweise gering [8.5].

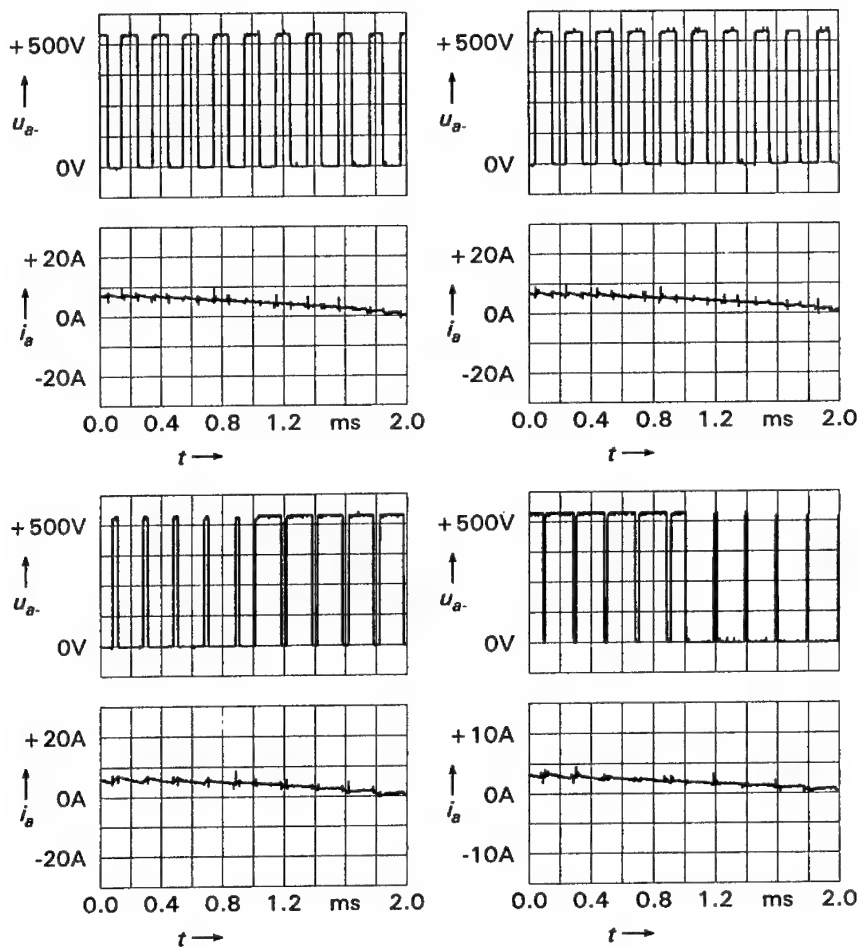


Bild 8.10: Verlauf der Spannung u_a und des Stroms i_a bei kleinem Modulationsgrad im Leerlauf ($M = 0.3$, $\alpha \approx 270^\circ$, $f_p = 5$ kHz)

Als Darstellung der exakten Verläufe zeigt Bild 8.10 Messungen entsprechend Bild 8.9 in 10-facher Auflösung, die etwa dem Zeitpunkt $t = 14$ ms in Bild 8.9 entsprechen bzw. dem Winkel $\alpha = \pi$ in Bild 8.6. Während in der oberen Reihe bei Sinusmodulation der Einzelphasen und Modulation mit symmetrischen Nullzuständen der Verlauf des Modulationsgrads einer Phase - und damit auch der Mittelwert der gepulsten Spannung - im dargestellten Bereich stetig

verläuft, sind in der unteren Reihe bei den zweiphasigen Modulationsverfahren deutlich die Sprünge im Spannungsverlauf zu erkennen.

8.4.3 Betrieb bei Begrenzung

8.4.3.1 Passive Begrenzung

Diese Art der Begrenzung ist eine Folge der in dieser Arbeit beschriebenen Steuerung (vgl. Abschnitt 5.7.5). Durch eine Stromregelung können ggf. Modulationsgrade m_a , m_b , m_c der Einzelphasen auftreten, die größer 1 sind. Die Folge ist, daß die daraus resultierenden Umschaltzeitpunkte auf einer der Grenzen oder außerhalb des Ausführungszeitraums liegen und daher unterdrückt werden müssen. Dadurch wird der entsprechende Modulationsgrad der Einzelphase auf den Wert 1 gesetzt. Das kann dazu führen, daß die Umschaltungen aller drei Phasen unterdrückt werden müssen. Das entspricht der Vollaussteuerung in der entsprechenden Pulsperiode. Die Bilder in 8.11 zeigen die Verläufe der Modulationsgrade m_a der Einzelphasen, wie sie sich unter Vorgabe eines Modulationsgrads M von 1.25 ohne passive Begrenzung ergeben würden. Die schattierten Gebiete stellen diejenigen Bereiche dar, in denen eine passive Begrenzung wirksam ist, d.h. $m_a > 1$. Sie wirkt sich in der Realität so aus, daß die grau schattierten Teile unberücksichtigt bleiben und dadurch der Modulationsgrad m_a der Einzelphasen konstant auf dem Wert 1 gehalten wird. Deutlich zu erkennen ist auch, daß trotz des Modulationsgrads M von 1.25 die Berechnung der Nullkomponente richtig durchgeführt wird, da die 60°- bzw. 30°-Bereiche, in denen m_a bei zweiphasigem Pulsen erster bzw. zweiter Art

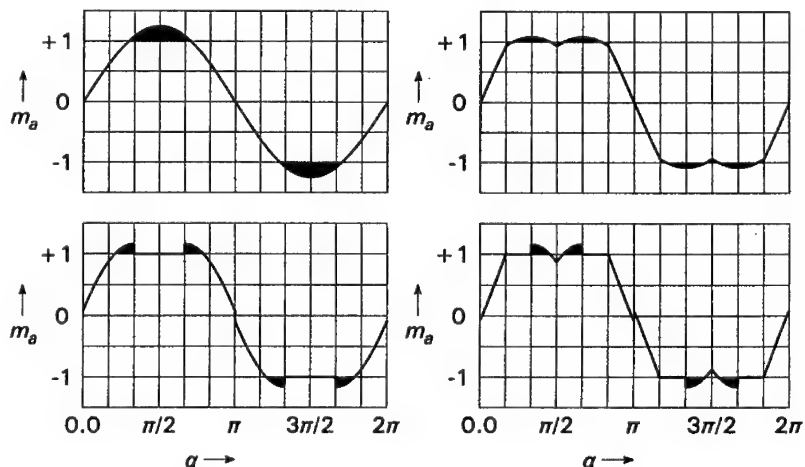


Bild 8.11: Verlauf des Modulationsgrads m_a der Phase a bei $M = 1.25$ (grau schattiert: Bereiche passiver Begrenzung)

durch die Nullkomponente auf dem Wert 1 gehalten wird, dieselben Bereiche sind wie in Bild 8.6.

Bild 8.12 zeigt die gemessenen Spannungs- und Stromverläufe, die sich bei einem Verlauf des Modulationsgrads der Einzelphasen nach Bild 8.11 unter Berücksichtigung der passiven Begrenzung ergeben. Es ist deutlich zu erkennen, daß bei allen vier Pulsverfahren zusätzliche Bereiche entstehen, in denen eine Phase überhaupt nicht mehr geschaltet wird. Das führt bei den zweiphasigen Verfahren dazu, daß entsprechend Bild 8.11 der Bereich, in dem $m_o \geq 1$ gilt,

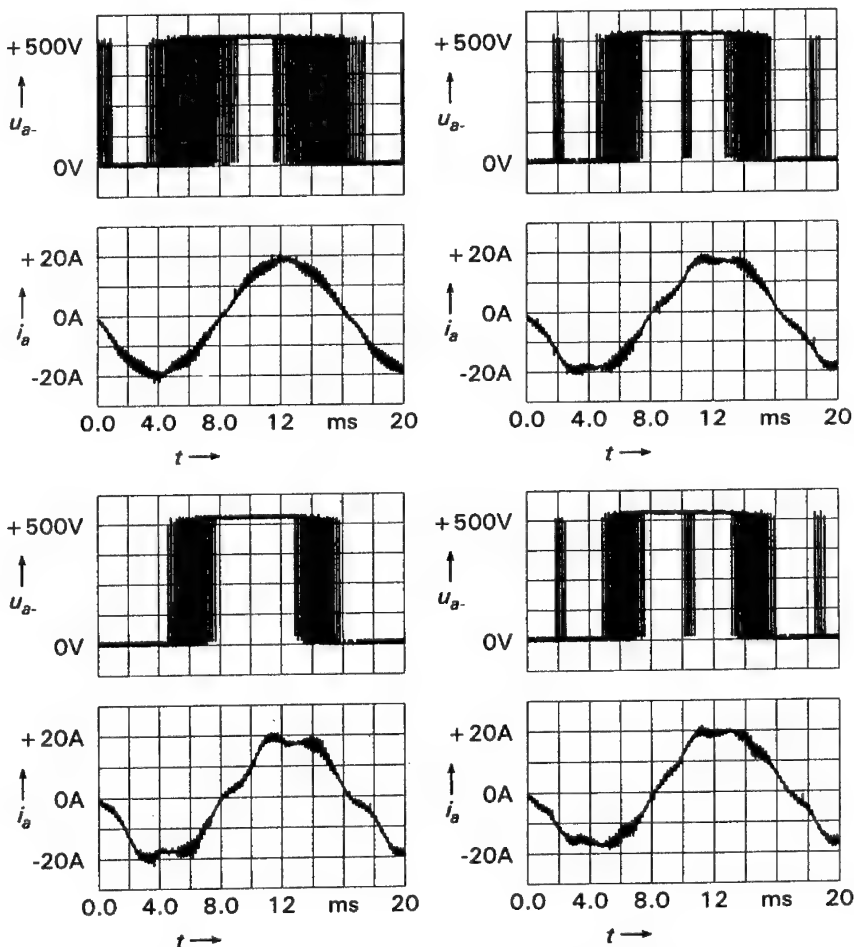


Bild 8.12: Strom- und Spannungsverläufe bei passiver Begrenzung ($M = 1.25, f = 60 \text{ Hz}$)

breiter wird. Die schmalen Bereiche der Spannungspulse, die sich bei Modulation mit symmetrischen Nullzuständen und bei zweiphasigem Pulsen zweiter Art ergeben (z.B. $t \approx 2.0$ ms oder $t \approx 10$ ms), würden bei weiterer Erhöhung des Modulationsgrads M immer weiter reduziert, sodaß sich die Spannungsverläufe aller vier Pulsverfahren immer mehr annäherten, und schließlich in der Vollaussteuerung mit 180° breiten Spannungsblöcken münden.

Durch die passive Begrenzung ist kein gleichmäßiger kreisförmiger Umlauf des sich ergebenden Spannungsraumzeigers mehr gegeben. Dadurch vergrößern sich die in der Literatur (z.B. [8.9]) beschriebenen typischen Oberschwingungen u.a. mit den Ordnungszahlen $v = -5$ und $v = 7$. Das ist vor allem in den Bildern der Pulsverfahren mit Nullkomponente gut ersichtlich.

8.4.3.2 Aktive Begrenzung

Die aktive Begrenzung nach Abschnitt 5.7.5 korrigiert die Umschaltzeiten T_j so, daß ein Spannungsraumzeiger entsteht, der genau auf der Linie des einschließenden Sechsecks der Stromrichter-Spannungszustände verläuft. Sie kann nur über Pulsverfahren mit Nullkomponente erreicht werden, indem man die Nullkomponente so bemißt, daß zwei Phasen gleichzeitig mit einem Modulationsgrad m von ± 1 für jeweils 120° betrieben werden, sodaß nur noch eine Phase geschaltet wird.

Nach den Gleichungen (5.21) bis (5.23) für die Nullkomponenten ergibt sich daraus immer eine der Nullkomponente entsprechende Verschiebung der Umschaltzeitpunkte von $T_{Null} = 0$. Dadurch stellt die Nullkomponente selbst keinen Freiheitsgrad mehr dar, anhand dessen sich die einzelnen Verfahren unterscheiden. Bild 8.13 zeigt die berechnete Kurve des Modulationsgrads m_a der Phase a, für den alle drei Verfahren mit Nullkomponente ganz ähnliche Verläufe ergeben. Zu beachten ist, daß im Bild die Flanken der trapezförmigen Kurve nicht exakt linear sind.

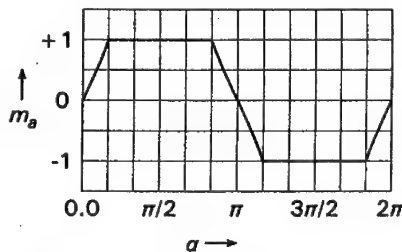


Bild 8.13: Verlauf des Modulationsgrads m für Verfahren mit Nullkomponente bei aktiver Begrenzung ($M = 1.5$)

Bild 8.14 zeigt die Spannungs- und Stromverläufe unter Vorgabe eines Modulationsgrads $M = 1.5$. Es stellen sich bei allen vier Pulsverfahren wieder zusätzliche Bereiche ein, in denen eine Phase nicht gepulst wird. Außerdem ist ebenfalls zu erkennen, daß die Spannungsverläufe der drei Verfahren mit Nullkomponente nahezu identisch sind. Das linke obere Bild in 8.14 entsteht bei Sinusmodulation der Einzelphasen. Bei diesem Verfahren wird zunächst die gleiche Korrektur angewandt. Die Nullkomponente entfällt jedoch, sodaß weiterhin Bereiche existieren, in denen $|m| \geq 1$ gilt. Dort wirkt automatisch die passive Begrenzung, sodaß sich

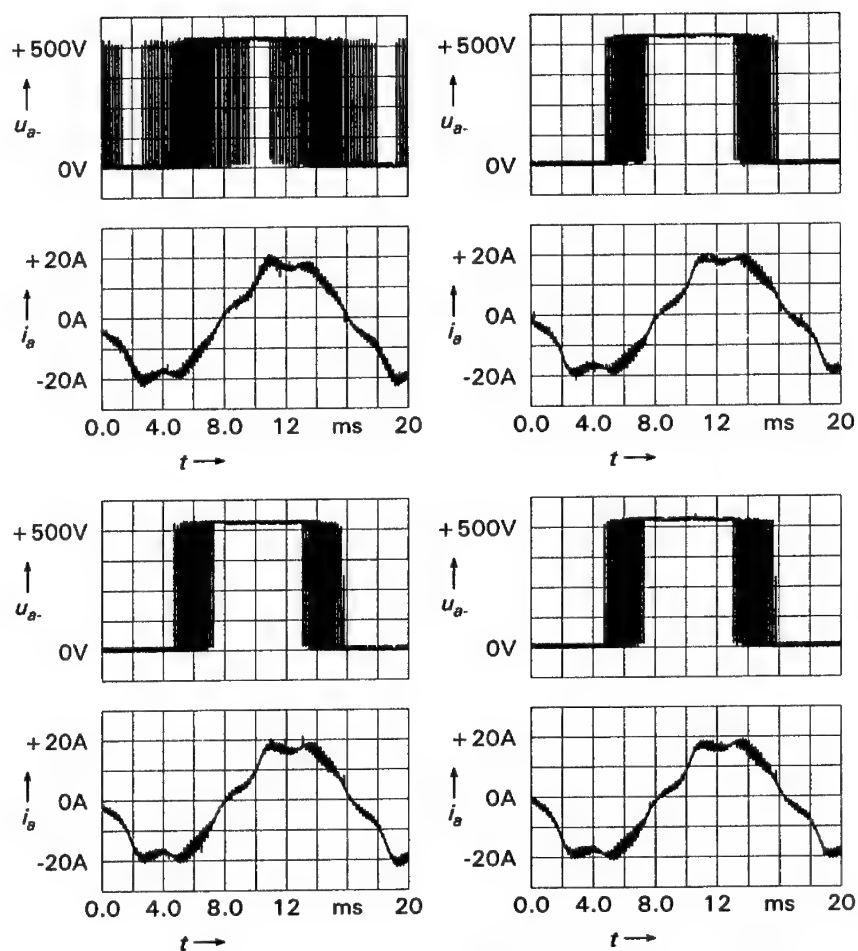


Bild 8.14: Strom- und Spannungsverläufe bei aktiver Begrenzung ($M = 1.5$, $f = 60$ Hz)

ein kleinerer Betrag des resultierenden Spannungsraumzeigers einstellt (vgl. Bild 8.12).

9 Zusammenfassung und Ausblick

In dieser Arbeit wird untersucht, ob und auf welche Weise mit einem PC ein Drehstromantrieb betrieben werden kann. Die Vorgaben dazu wären die Benutzung eines populären Betriebssystems, die Ausführung der Programmierarbeiten in einer Hochsprache sowie die Verwendung von handelsüblichen PC-Komponenten. Die Arbeit zielt auf mögliche Einsatzgebiete in der Forschung und Entwicklung sowie Lehre ab. Der industrielle Einsatz wird aufgrund der vergleichsweise hohen Kosten der Hardware auf wenige Nischenanwendungen begrenzt bleiben.

Die Erstellung eines geeigneten Steuerungskonzeptes setzte Untersuchungen zu den einzelnen Hardwarekomponenten und zur Anwendbarkeit von Steuermethoden voraus. Daraus wurden insbesondere für die interruptbetriebene Stromrichtersteuerung wichtige Erkenntnisse gewonnen, die in die Gestaltung der Halbperiodensteuerung eingeflossen sind. Eine solche Interruptsteuerung ist zur Zeit nur mit dem Betriebssystem DOS realisierbar, da nur mit diesem aufgrund seiner Single-Tasking-Eigenschaft akzeptable Zeiten für die Interruptverarbeitung zu erreichen sind und es damit echtzeitfähig ist. Es ist allerdings anzumerken, daß durch die Verwendung von DOS die Leistungsfähigkeit heutiger CPU bzw. PC bei weitem nicht ausgeschöpft wird. Dennoch war es nicht notwendig einzelne Teile des Quellcodes in Maschinensprache zu programmieren, sodaß die Vorteile der Hochsprachenprogrammierung voll genutzt werden konnten.

Durch den internen Aufbau eines PC müssen im Vergleich zu Mikrocontrollern zusätzliche Aspekte bei der Interruptverarbeitung berücksichtigt werden, welche die vielfältigen Möglichkeiten eines PC-Systems einschränken. So sind Vorgänge zur Ein- oder Ausgabe von Daten und insbesondere solche Vorgänge, die von Interrupts gesteuert werden, nach wie vor äußerst komplex und mit einem vergleichsweise hohen Zeitbedarf behaftet, sodaß sie sich ggf. gegenseitig blockieren können. Dadurch ist bei solchen Vorgängen nicht mehr die Annahme gültig, daß sie in vernachlässigbar kurzer Zeit bearbeitet werden. In dieser Arbeit wurde gezeigt, daß durch diese beiden Gruppen von Vorgängen sowie den grundlegenden Berechnungsroutinen ein Grundzeitbedarf innerhalb eines Berechnungszyklus entsteht, der nur bedingt verringert werden kann. Beides ist aber für Stromrichtersteuerungen unbedingt notwendig. Deshalb wurden die rechnerinternen Abläufe bei Interrupts und deren Zeitbedarf eingehend untersucht. Aus den Ergebnissen wurden mehrere Kalibrierrountinen entwickelt, die dazu dienen, vor dem eigentlichen Stromrichterbetrieb wichtige hardwareabhängige Parameter (z.B. Interruptzeit

T_{ini}) zu ermitteln, die dann im Betrieb berücksichtigt werden. Dadurch wird die Steuerung selbst hardwareunabhängig und kann im Prinzip ohne Änderungen auf unterschiedlich konfigurierten Systemen arbeiten. Obwohl sich der Grundzeitbedarf dadurch nicht ändert, ist es auf diese Weise möglich, beispielsweise Verzögerungen erster Art voll zu kompensieren und damit Stromverzerrungen entscheidend zu verringern. Der Grundzeitbedarf bleibt daher ein Maß für die erzielbare maximale Pulsfrequenz. Mit der beschriebenen Hard- und Software wurde eine Pulsfrequenz von 5 kHz bei Halbperiodensteuerung, d.h. einem Berechnungszyklus entsprechend 10 kHz, ohne Schwierigkeiten erreicht.

In Abschnitt 5.5.2 werden zwei Methoden vorgestellt, welche die Auswirkungen des Zeitbedarfs von Interruptroutinen in deterministischen Prozessen weitgehend kompensieren. Die erste davon ist die Vereinigungsmethode, die zwei zeitlich nahe beieinander liegende Phasenumschaltungen gleichzeitig in einem Interrupt ausführt. Die andere ist die Warteschleifenmethode, die einen variablen, kurzen Zeitraum durch die Ausführung einer festgelegten Anzahl an Null-Operationen überbrückt. Aufgrund der höheren Flexibilität und der besseren Ergebnisse im Betrieb der Warteschleifenmethode wurde die Vereinigungsmethode jedoch nicht weiter verfolgt.

Für die Anwendung der Halbperiodensteuerung auf unterschiedliche Pulsverfahren stellte sich die in dieser Arbeit entwickelte Umschaltunterdrückung als eine wichtige Methode heraus, die durch die hier genannte "passive Begrenzung" gleichzeitig den stabilen Betrieb des Stromrichters bei möglichen Übersteuerungen aufrecht erhält. Der entscheidende Gedanke war die Entkopplung von aufeinander folgenden Berechnungszeiträumen bzgl. der Stromrichterzustände. Das wurde durch die Einführung des vom Normalzustand z_{norm} ggf. verschiedenen Anfangszustands z_{Anfang} erreicht.

Eine Strom- bzw. Maschinenregelung wurde in dieser Arbeit nicht realisiert; die zum Einbau einer solchen Regelung notwendigen Softwarestrukturen - wie z.B. die Ermittlung und Verwendung von Zeitfenstern - wurden jedoch vorbereitet. Das beinhaltet auch zusätzliche Elemente in der Datenstruktur eines Steuereingriffs bzw. ein Variablenfeld, in die beispielsweise nicht kompensierbare Fehlerzeiten bei den Phasenumschaltungen eingetragen und zu einem späteren Zeitpunkt berücksichtigt werden können.

Auf dem Gebiet der Bauelemente und PC-Komponenten geht der Trend immer noch eindeutig hin zu kleineren Halbleiterstrukturen und damit auch zu höheren Taktraten. So sind bereits heute Prozessoren mit einer Taktrate von 600 MHz auf dem Markt erhältlich, während die Messungen aus Abschnitt 8 noch auf einem PentiumII-System mit 400 MHz durchgeführt wurden. Neue Strukturen sowohl im Prozessor wie auch im On-Chip-Cache-Speicher (L1-

Cache) tragen ebenfalls zu einer höheren Leistungsfähigkeit bei. Einen anderen Weg beschreibt die immer weitergehende Systemintegration. So sind heute bereits integrierte Bausteine erhältlich, die einen kompletten PC mit CPU, Speicher- und Interruptcontroller sowie den notwendigen Peripheriecontrollern beispielsweise für Graphikkarte oder Schnittstellen enthalten [9.1]. Mit solchen Einheiten kann der Datenaustausch oder die Erfassung von Interrupts sehr effektiv ausgeführt werden, sodaß sich ein geringerer Grundzeitbedarf ergibt.

Im Bereich der Betriebssysteme wird die Entwicklung sicher in starkem Maße von den zukünftigen Anwendungsgebieten der PC abhängig bleiben. Da in nahezu allen herkömmlichen Anwendungen keine strenge Echtzeit gefordert wird, ist nicht zu erwarten, daß populäre Betriebssysteme gerade dies erfüllen werden. Ein großer Sprung bezüglich der Anwendung von PC als Steuergerät in der Antriebstechnik ist aber nur dann zu erwarten, wenn 32-Bit-Betriebssysteme mit kurzen Interruptzeiten zur Verfügung stehen. In der letzten Zeit sind vor allem auf dem Gebiet der sogenannten "embedded systems" hinsichtlich der Echtzeitfähigkeit zahlreiche Entwicklungen entstanden, die sich auch auf den PC-Bereich auswirken könnten.

Die Aussicht auf zukünftige Entwicklungen führt auch auf Möglichkeiten der Weiterentwicklung des vorgestellten Antriebssystems. Mit den vorgegebenen Software-Strukturen kann ein geeignetes Regelungs- und Kommunikationskonzept entwickelt werden, was durchaus die Realisierung eines Vier-Quadranten-Antriebs ermöglicht. Mit leistungsfähigeren Systemen ist auch die Programmierung im Protected Mode von DOS sinnvoll, sodaß dessen Vorteile (z.B. Zugriff auf Speicherbereiche oberhalb 1 MB) hinsichtlich der Simulation des Echtzeitbetriebs voll genutzt werden können. Es bleibt aber festzustellen, daß mit 16-Bit-Betriebssystemen immer nur ein Teil der Systemleistung genutzt werden kann. Deshalb könnte als Ergänzung zu dieser Arbeit geprüft werden, ob es möglich ist, ein 32-Bit-Betriebssystem so zu konfigurieren, daß beispielsweise die Stromrichtersteuerung als einzige Anwendung läuft. Neben der Nutzung der gesamten Systemleistung könnten sich dadurch ebenfalls konstante und ausreichend kurze Interruptzeiten ergeben. Es ist dazu eine grundlegende und detaillierte Einarbeitung in das gewählte Betriebssystem und die Programmierung der entsprechenden Schutzmechanismen des Protected Mode erforderlich.

10 Literaturverzeichnis

Literaturangaben zu Abschnitt 1:

- [1.1] MarketLine International Ltd.: PC-Based Industrial Control Systems; Executive Summary, MarketLine International Ltd., 16 Connaught St., London, UK, 1998.
- [1.2] S.R. Bowes, Y.S. Lai: Investigation into Optimising High Switching Frequency regular sampled PWM Control for Drives and Static Power Converters; IEE Proceedings of Electrical Power Applications, Vol. 143, No. 4, July 1996.
- [1.3] D. Vincenti, P.D. Ziogas, R.V. Patel: A PC-Based Pulse-Width Modulator for Static Converters; IEEE Transactions on Industrial Electronics, Vol. 37, No. 1, February 1990.
- [1.4] S.R. Bowes, P.R. Clark: Simple Microprocessor Implementation of New Regular-Sampled Harmonic Elimination PWM Techniques; IEEE Transactions on Industry Applications, Vol. 28, No. 1, January/February 1992.
- [1.5] M.N. Anwar, Md. B. Uddin, M.A. Choudhury: Microcomputer Based On-line Control of a Delta Modulated Sine PWM Inverter; 7th European Conference on Power Electronics and Applications (EPE97), Trondheim, Norwegen, 1997.
- [1.6] K. Nagaswamy, R. Oruganti, L.K. Sang: Implementation of Predicted (on-time) Equal Charge Criterion Control of a single phase oost AC-DC Converter; Proceedings of the IEEE International Conference on Power Electronics and Drive Systems, S. 494 ff., Singapore, 1997.

Literaturangaben zu Abschnitt 2:

- [2.1] A. Bode, W. Händler: Rechnerarchitektur, Grundlagen und Verfahren; S. 196 ff., Springer-Verlag Berlin Heidelberg New York, 1980.

-
- [2.2] K. Brand: Entwicklungsprojekte mit Mikroprozessoren; Abschnitt 2.4, S. 69 ff., Hüthig Buch Verlag, Heidelberg, 1989.
- [2.3] Intel: 80286:Hardware Reference Manual; Intel Corporation, 1987
- [2.4] Intel Corp., et al.: Universal Serial Bus Specification; Revision 1.0, Intel Corporation, 1996.
- [2.5] Intel corp.: Pentium Prozessor Family Developer's Manual, Vol.3: Architecture and Programming Manual; Intel Corp. Mt. Prospect, IL, USA, 1995.
- [2.6] S. Fedtke: 80286 / 80386 / i486 effizient programmiert, Bd.1: Vom Prozessorkonzept zur Programmierung; Abschnitt 4.1, S. 47, Friedr. Vieweg & Sohn Verlagsgesellschaft, Braunschweig, 1991.
- [2.7] H.-P. Messmer: PC-Hardwarebuch; Abschnitt 4.3, S. 188 ff., 3. Auflage, Addison-Wesley GmbH, 1995.
- [2.8] S. Fedtke: 80286/80386/i486 effizient programmiert, Bd.1: Vom Prozessorkonzept zur Programmierung; Abschnitt 11.3., S. 254 ff., Vieweg Verlag, Braunschweig, 1991.
- [2.9] S. Fedtke: 80286/80386/i486 effizient programmiert, Bd.1: Vom Prozessorkonzept zur Programmierung; Abschnitt 11.3., S. 264 ff., Vieweg Verlag, Braunschweig, 1991.
- [2.10] Intel corp.: Using the 8259A Programmable Interrupt Controller; Application Note AP59, 1979.
- [2.11] H.-P. Messmer: PC-Hardwarebuch; Abschnitt 23.5, S. 634 ff., 3. Auflage, Addison-Wesley GmbH, 1995.
- [2.12] Intel Corp.: 82C54 CHMOS Programmable Interval Timer; Data Sheet, 1993.
- [2.13] H.-P. Messmer: PC-Hardwarebuch; Abschnitt 24.3, S. 653 ff., 3. Auflage, Addison-Wesley GmbH, 1995.
- [2.14] H.-P. Messmer: PC-Hardwarebuch; Abschnitt 25.2.3, S. 684 ff., 3. Auflage, Addison-Wesley GmbH, 1995.

- [2.15] E. Solari, G. Willse: PCI - Hardware and Software; S. 803 ff., Annabooks, San Diego, Ca., USA, 1995.
- [2.16] Intel Corp., et al.: PC 99 System Design Guide: A Technical Reference for Designing PCs and Peripherals for the Microsoft Windows Family of Operating Systems; Preview draft, Version 0.3, Intel Corporation und Microsoft Corporation, 1998.
- [2.17] Intel Corp.: 82C55A Programmable Peripheral Interface; Data Sheet, Intel Corporation, USA, 1993.
- [2.18] H.P. Messmer: PC-Hardwarebuch; Abschnitt 31, S. 1009 ff., Addison-Wesley GmbH, 1995.
- [2.19] c't Computermagazin: Glossar Cache- und Speichertechnik; c't Computermagazin, Ausgabe 2/96, S. 218, Heise-Verlag, 1996.
- [2.20] Intel Corp.: Pentium Processors Family Developer's Manual, Vol. 1: Pentium Processors; S. 3-15 ff., Intel Corporation, Mt. Prospect, Il., USA, 1995.
- [2.21] H.-P. Messmer: PC-Hardwarebuch; Abschnitt 12.3.8, S. 384, 3.Auflage, Addison-Wesley GmbH, 1995.
- [2.22] J.F. Wollert: Doping für den Betriebssystem-Kern; Zeitschrift "Elektronik", Ausgabe 20/97, S. 46-59, WEKA Fachzeitschriften-Verlag, Poing, 1997.
- [2.23] QNX Software Systems: QNX Betriebssystem: System Architektur; QNX Software Systems Ltd., Ontario, Kanada, 1997.

Literaturangaben zu Abschnitt 3:

- [3.1] SEMIKRON INTERNATIONAL: Semikron Leistungselektronik; Datenbuch 1997/98, S. B11-47 ff., Semikron International, Nürnberg, 1996.
- [3.2] SEMIKRON INTERNATIONAL: Semikron Leistungselektronik; Datenbuch 1997/98, S. B6-53 ff., Semikron International, Nürnberg, 1996.

- [3.3] SEMIKRON INTERNATIONAL: Semikron Leistungselektronik; Datenbuch 1997/98, S. B14-21 ff., Semikron International, Nürnberg, 1996.
- [3.4] SEMIKRON INTERNATIONAL: Semikron Leistungselektronik; Datenbuch 1997/98, S. B14-28, Semikron International, Nürnberg, 1996.
- [3.5] U. Post: Universalgenies: x86-Prozessoren für jeden Bedarf, c't Computermagazin, Heft 10/98, S. 196 ff., Heise-Verlag, 1998.
- [3.6] S.R. Bowes, P.R. Clark: Transputer-Based Optimal PWM Control of Inverter Drives; IEEE Transactions on Industry Applications, Vol. 28, No. 1, January/February 1992
- [3.7] U. Post, G. Schnurer: Tempo 400: 18 BX-Boards für Intels neuen Pentium-II-Prozessor, c't Computermagazin, Ausgabe 8/98, S. 166 ff, Heise-Verlag, 1998.
- [3.8] Kernighan, Ritchie: Programmieren in C; S. 3ff., 2.Auflage, Hanser Verlag, 1990.

Literaturangaben zu Abschnitt 4:

- [4.1] S.R. Bowes, Y.S. Lai: Investigation into Optimising High Switching Frequency regular sampled PWM Control for Drives and Static Power Converters; S. 281 ff., IEE Proceedings on Electrical Power Applications, Vol. 143, No. 4, July 1996.
- [4.2] F. Jenni, D. Wüest: Steuerverfahren für selbstgeführte Stromrichter; S. 116, Hochschulverlag an der ETH Zürich und B.G. Teubner Stuttgart, Zürich, Stuttgart, 1995.
- [4.3] F. Jenni, D. Wüest: Steuerverfahren für selbstgeführte Stromrichter; S. 111 ff., Hochschulverlag an der ETH Zürich und B.G. Teubner Stuttgart, Zürich, Stuttgart, 1995.
- [4.4] D. Leggate, R.J. Kerkman: Pulse-Based Dead-Time Inverter Compensator for PWM-Voltage Inverters; S. 191 ff., IEEE Transactions on Industrial Electronics, Vol. 44, No. 2, April 1997.
- [4.5] W. Boeck, et al.: Hochspannungstechnik, Springer-Verlag, New York, München.
- [4.6] K.P. Kovács: Symmetrische Komponenten in Wechselstrommaschinen; Birkhäuser Verlag, Basel-Stuttgart, Schweiz, 1962.

- [4.7] G. Müller: Theorie elektrischer Maschinen; Abschnitt 12.7, S. 310 ff., VCH Verlagsgesellschaft, Weinheim, Deutschland, 1994.
- [4.8] F. Jenni, D. Wüest: Steuerverfahren für selbstgeführte Stromrichter; Abschnitt 3, S. 60 f., vdf Hochschulverlag AG und Teubner Verlag Stuttgart, Zürich, Stuttgart, 1995.
- [4.9] D. Dietrich, W. Kohnhäuser: Mikrocomputergeregelte Asynchronmaschinen; S. 30, Oldenbourg Verlag München, Deutschland, 1986.
- [4.10] L. Abraham, R. Blümel: Optimization of three phase Pulse Pattern by variable Zero Sequence Component; Conference on European Power Electronics 1991 (EPE91), Florenz, 1991.
- [4.11] F. Jenni, D. Wüest: Steuerverfahren für selbstgeführte Stromrichter; Abschnitt 7, S. 138, vdf Hochschulverlag AG und Teubner Verlag Stuttgart, Zürich, Stuttgart, 1995.
- [4.12] F. Jenni, D. Wüest: Steuerverfahren für selbstgeführte Stromrichter; Abschnitt 5, S. 85 f., vdf Hochschulverlag AG und Teubner Verlag Stuttgart, Zürich, Stuttgart, 1995.
- [4.13] J. Holtz: Pulswidth Modulation - A Survey; IEEE Transactions on Industrial Electronics, S.410 ff., Vol. 39, No. 5, Dec. 1992.
- [4.14] S.R. Bowes, Y.S. Lai: The Relationship between Space-Vector Modulation and Regular-Sampled PWM; IEEE Transaction on Industrial Electronics, S. 670 ff., Vol. 44, No. 5, Oct. 1997.
- [4.15] A.M. Trzynadlowski, R.L. Kirlin, S.F. Legowski: Space Vector PWM Technique with Minimum Switching Losses and Variable Puls Rate; IEEE Transactions on Industrial Electronics, S. 173 ff., Vol. 44, No. 2, Apr. 1997.
- [4.16] F. Jenni, D. Wüest: Steuerverfahren für selbstgeführte Stromrichter; Abschnitt 8.2, S. 160 ff., vdf Hochschulverlag AG und Teubner Verlag Stuttgart, Zürich, Stuttgart, 1995.

Literaturangaben zu Abschnitt 5:

- [5.1] Stephen Fedtke: 80286/80386/i486 effizient programmiert, Bd.2: AT-Betriebssysteme; Anhang A.1, S. 471, Friedr. Vieweg & Sohn Verlagsgesellschaft, Braunschweig, 1991.
- [5.2] W. Frank: PWM-Inverter Drive Control operating with a standard Personal Computer; Proceedings of the 1997 IEEE Conference on Power Electronics and Drive Systems, Singapur, 1997
- [5.3] SEMIKRON INTERNATIONAL: Semikron Leistungselektronik, Datenbuch 1997/98; S. B14-32, Semikron International, Nürnberg, 1996.
- [5.4] A.M. Trzynadlowski, R.L. Kirlin, S.F. Legowski: Space Vector PWM Technique with Minimum Switching losses and Variable Pulse Rate; IEEE Transactions on Industrial Electronics, S. 173 ff., Vol. 44, No. 2, Apr. 1997.
- [5.5] H. van der Broeck: Analysis of the Harmonics in Voltage Fed Inverter Drives Caused by PWM schemes with Discontinuous Switching Operation; Proceedings of 4th European Conference on Power Electronics and Applications 1991 (EPE91), S. 3-261 Florenz, 1991.
- [5.6] Computer Boards Inc.: CIO-DAS 1600 User's Manual, Rev. 3; S. 6, Computer Boards Inc., Mansfield (MA.), USA, 1994.
- [5.7] H.-P. Messmer: PC-Hardwarebuch; Abschnitt 24.5, S. 655 ff., 3.Auflage, Addison-Wesley.

Literaturangaben zu Abschnitt 6:

- [6.1] S. Fedtke: 80286/80386/i486 effizient programmiert, Bd. 2: AT-Betriebssysteme; Abschnitt 2.14, S. 426, Vieweg Verlag, Braunschweig, 1991.
- [6.2] S. Fedtke: 80286/80386/i486 effizient programmiert, Bd. 2: AT-Betriebssysteme; Abschnitt 2.7.2, S. 270, Vieweg Verlag, Braunschweig, 1991.

- [6.3] H.-P. Messmer: PC-Hardwarebuch; Abschnitt 32, S. 1047 ff., 3.Auflage, Addison-Wesley.

Literaturangaben zu Abschnitt 7:

- [7.1] SEMIKRON International: Semikron, Datenbuch für Leistungselektronik; S. B-34, Semikron, Deutschland, 1996.

Literaturangaben zu Abschnitt 8:

- [8.1] Hewlett-Packard Company: Programming Reference: HP 54505B, HP 54506B, HP 54510B and HP 54512B Digitizing Oscilloscopes; Hewlett-Packard, P.O. Box 2197, Colorado Springs, CO 80901, USA.
- [8.2] Pewatron: Scope Differential Probe, SI-9000; Datenblatt, Pewatron AG, Wallisellen/Zürich, Schweiz.
- [8.3] Tektronix Inc.: A6303 Current Probe; Instructions, Tektronix. Inc., Beaverton, Oregon 97077, USA.
- [8.4] Tektronix Inc.: AM 503 Current Probe Amplifier; Instruction Manual, Tektronix Inc., Beaverton, Oregon 97077, USA.
- [8.5] L.Abraham, R. Blümel: Optimization of three phase Pulse Pattern by variable Zero Sequence Component; Proceedings of 4th European Conference on Power Electronics 1991 (EPE91), S. 3-272, Florenz, 1991.
- [8.6] H. van der Broeck: Analysis of the Harmonics in Voltage Fed Inverter Drives Caused by PWM schemes with Discontinuous Switching Operation; Proceedings of 4th European Conference on Power Electronics and Applications 1991 (EPE91), S. 3-261 Florenz, 1991.
- [8.7] F. Jenni, D. Wüest: Steuerverfahren für selbstgeführte Stromrichter; Abschnitt 7.1, S. 110 f., vdf Hochschulverlag AG und Teubner Verlag Stuttgart, Zürich, Stuttgart, 1995.

[8.8] Gernar Müller: Theorie elektrischer Maschinen; Abschnitt 11.1, S. 277 ff., VCH Verlagsgesellschaft, Weinheim, Deutschland, 1995.

[8.9] F. Jenni, D. Wüest: Steuerverfahren für selbstgeführte Stromrichter; Abschnitt 6.3, S. 104 f., vdf Hochschulverlag AG und Teubner Verlag Stuttgart, Zürich, Stuttgart, 1995.

Literaturangaben zu Abschnitt 9:

[9.1] STMicroelectronics: High Performance Industrial PC on a Chip; Product Preview, STMicroelectronic.

Literaturangaben zum Anhang :

[A.1] S. Fedtke: 80286/80386/i486 effizient programmiert, Bd.2: AT-Betriebssysteme; Abschnitt A.1., S. 470 ff., Vieweg Verlag, Braunschweig, 1991.

11 Verzeichnis der Abkürzungen und Indizes

Bauelemente:

Bezeichnung von Schaltern mit aufrechten Buchstaben, z.B. T4, alle anderen Bauelemente mit kursiven Buchstaben R_A !

a, b, c . Kennzeichnung der Phasen eines Drehstromsystems	M Mittelpunkt der Asynchronma- schine
A, B, C Halbbrücken, Brückenzeige.	R Widerstand
C Kapazität	T Transistor
D Diode	
L Induktivität	

Physikalische Größen / allg. Variablen:

Kleingeschriebene Größen stellen zeitabhängige Größen dar (z.B. $i_{Tl} = i_{Tl}(t)$)

Großgeschriebene Größen stellen zeitlich konstante Größen oder Effektivwerte dar (z.B. U_d)

A Aussteuerungsgrad	\underline{e} komplexer Raumzeiger der Gegenspannung
K Korrekturfaktor bei Begrenzung	f Frequenz
M Modulationsgrad	i Strom
Im Imaginärteil	\underline{i} komplexer Raumzeiger des Stroms
Re Realteil	j Nummer eines Steuereingriffs
SE Steuereingriff	m Modulationsgrad einer Phase
T Zeitdauern, Intervalle	sw_red Anzahl der Vereinigungen bzw. der Umschaltunterdrückungen
T_{Aus} . . . Interrupt-Aussprunzeit	t Zeit
T_B Interrupt-Bearbeitungszeit	u Spannung
T_{Ein} . . . Interrupt-Einsprunzeit	\underline{u} komplexer Raumzeiger der Spannung
T_{Int} Interruptzeit	z Stromrichterzustand
T_{min} . . . Mindestzeit für Verz. 2.Art	
a Art eines Steuereingriffs	
\underline{a} komplexer Dreher ($=e^{j2\pi/3}$)	
e induzierte Gegenspannung	

Δ Änderung, Differenz
 α Winkel
 ω Kreisfrequenz

v Ordnungszahl von Oberschwingungen

Indizes:

A, B .. Halbperiodentypen
 A Ausgang (z.B. $u_A(t)$)
 AZ ... Ausführungszeitraum
 $Anfang$ Anfangsgröße
 BZ ... Berechnungszeitraum
 E Eingang (z.B. $u_E(t)$)
 F Fehlergröße (z.B. T_F)
 $Fehler$ Fehlergröße
 H Halbperiode
 L lastseitig (z.B. $i_L(t)$)
 M bezogen auf Mittelpunkt der ASM
 $Null$.. Nullkomponentengröße
 V Verzögerung
 a, b, c . Phasenbezeichnungen
 art ... Art eines Steuereingriffs
 d Zwischenkreis (z.B. U_d)
 h harmonisch
 j Laufindex für sortierte Umschaltzeitpunkte
 i Laufindex für unsortierte Umschaltzeitpunkte

i Nummer einer Halperiode
 k, l, n . Laufindizes allgemein
 max .. Maximalgröße
 min ... Minimal- oder Mindestgröße
 $norm$. Norm- oder Normalgröße
 out ... Ausgang (meist zum Stromrichter hin)
 p gepulst
 w Führungsgröße
 $wart$.. Warteschleife
 α Realteil
 β Imaginärteil
 0 nullpunktbezogen
 0 Nullkomponente
 $0, 1, 2...$ Nummern der sortierten Steuereingriffe
 $1, 2, 3,..$ Nummern der unsortierten Steuereingriffe
 1 Grundschiwingung

Hochzeichen

* diskretisierte Größe
' kennzeichnet neu berechnete Größen, die im Programm die Größe ohne Hochzeichen ersetzt

$0, 1, 2$. Nummer der zu einem Stromrichterzustand gehörigen Größe

A PC- und Prozessorinterna

A.1 Registersatz

In der Hochsprachenprogrammierung bleiben die Register selbst meist unberücksichtigt und werden nur selten direkt programmiert. Allein beim direkten Aufruf von Software-Interrupt-routinen müssen entsprechende Registervariablen mit Werten belegt werden, damit der Interrupt ordnungsgemäß ausgeführt werden kann.

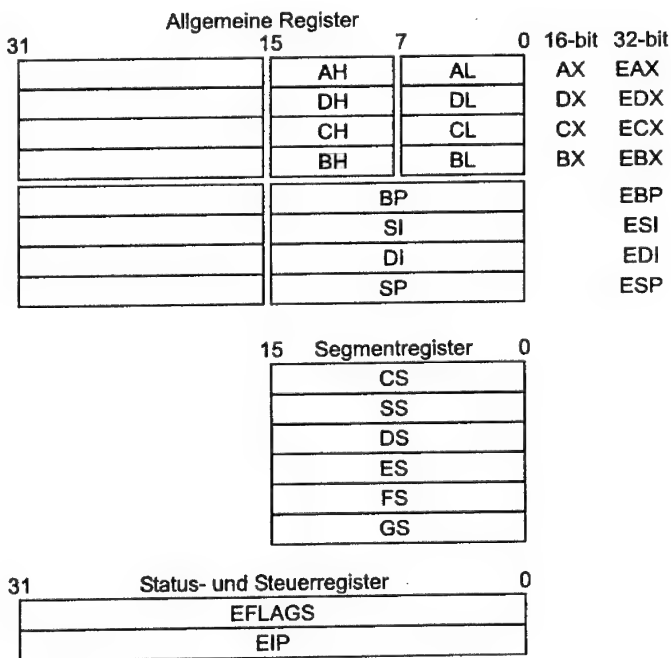


Bild A.1: Registersatz eines Pentium-Prozessors

Prozessoren der Intel-Familie (und deren Derivate) verfügen modellunabhängig über die gleiche Anzahl an Registern. Diese unterscheiden sich allerdings durch die Registerbreite. So verfügen beispielsweise Modelle ab der Serie 80386 über Register mit einer Breite von 32-bit. Obwohl Pentium-Prozessoren Busse mit einer Breite von 64-bit bedienen können, sind deren

Register ebenfalls nur 32 Bit breit. Bild A.1 zeigt den Registersatz eines Pentium-Prozessors.
Die Registerbezeichnungen bedeuten im einzelnen:

EAX: keine spezielle Bedeutung
EBX: keine spezielle Bedeutung
ECX: keine spezielle Bedeutung
EDX: keine spezielle Bedeutung
EBP: Base-Pointer (Basis-Zeiger)
ESI: Source-Index (Index des Segments des Quelloperanden)
EDI: Destination-Index (Index des Segments des Zieloperanden)
ESP: Stackpointer (Stapelzeiger)
CS: Code-Segment
SS: Stack-Segment
DS: Data-Segment
ES: E-Space-Segment
FS: keine weitere Bedeutung
GS: keine weitere Bedeutung

Die frei verwendbaren Register EAX, EBX, ECX und EDX sind aus Kompatibilitätsgründen wiederum unterteilt in 16-bit Register (AX, BX, CX, DX) bzw. 8-bit Register (AL, AH, BL, etc.). Wird für die Programmierung nicht die gesamte Registerbreite benutzt, so bleiben die restlichen Bits (z.B. die Bits 16-31) unberücksichtigt. Während die Register EAX, EBX, ECX und EDX meist Daten enthalten, dienen die Register EBP, ESI, EDI und ESP der Adreßrechnung.

A.2 Belegung der Interruptleitungen

Im folgenden ist die Belegung der Interrupt-Request-Leitungen angegeben, wie sie bei Standard-PC auftreten. Im Prinzip kann davon jede Interruptleitung umbelegt werden, für den Normalbetrieb ist es jedoch sinnvoll zusätzliche Geräte nur auf die freien Interrupts zu konfigurieren, da sonst Gerätekonflikte auftreten können.

Leistungsnummer	Belegung
IRQ 0	Systemtimer 0
IRQ 1	Tastatur
IRQ 2	Anschluß des zweiten PIC

IRQ 3	COM1
IRQ 4	COM2
IRQ 5	LPT2
IRQ 6	Diskette
IRQ 7	LPT1
IRQ 8	Echtzeituhr
IRQ 9	Umleitung auf IRQ 2
IRQ 10	frei
IRQ 11	frei
IRQ 12	frei
IRQ 13	numer. Co-Prozessor
IRQ 14	Festplatte
IRQ 15	frei

A.3 Interrupt-Descriptor-Table (IDT)

Die IDT ist sozusagen ein Wegweiser, der angibt, welche Routine im Falle eines Interrupts bearbeitet werden soll. Sie besteht aus 256 Einträgen zu je 2 x 16-bit. Ein 16-bit-Wert wird in das IP-Register geladen und der andere in das CS-Register. Zusammen bilden sie die Startadresse der auszuführenden Interrupt-Service-Routine. Sie kann sowohl in Speicherbereich des BIOS liegen wie auch in jedem anderen möglichen Speicherbereich. Das Programmieren und Einfügen individueller Interruptroutinen ist letztlich das Umbelegen von IDT-Einträgen. Eine Liste der wichtigsten Einträge ist der Literatur (z.B. [A.1]) zu entnehmen.

A.4 Scancode der Tastatur

Der ist Scancode tastatur- und länderunabhängig. Der Tastaturcontroller erzeugt jeweils für das Drücken bzw. Loslassen derselben Taste einen unterschiedlichen Scancode (Makecode bzw. Breakcode). Der Breakcode ergibt sich dabei durch das Erhöhen des jeweiligen Makecodes um 128. In der folgenden Tastatur sind diejenigen Scancodes angegeben, die durch das Drücken der entsprechenden Taste einer deutschen Tastatur (MF-II) entstehen.

Taste	Scancode
ESC	1
F1	59
F2	60
F3	61
F4	62
F5	63
F6	64
F7	65
F8	66
F9	67
F10	68
F11	87
F12	88
^	41
1	2
2	3
3	4
4	5
5	6
6	7
7	8
8	9
9	10
0	11
ß	12
'	13
Löschen	14

Tabulator	15
q	16
w	17
e	18
r	19
t	20
z	21
u	22
i	23
o	24
p	25
ü	26
'+'	27
CapsLock	58
a	30
s	31
d	32
f	33
g	34
h	35
j	36
k	37
l	38
ö	39
ä	40
#	43
Enter	28
Shift li.	42

<	86
y	44
x	45
c	46
v	47
b	48
n	49
m	50
,	51
.	52
-	53
Shift re.	54
Strg li.	29
Alt	56
Leertaste	57
Alt gr.	56
Strg re.	29
Einfügen	82
Entfernen	83
Pos. 1	71
Ende	79
Bild ↑	73
Bild ↓	81
Druck	55
Rollen	70
Pause	29+69
↑	72
←	75

↓	80
→	77
Num. ON	69
/	53
'*'	55
'_'	74
'+'	78

Enter	28
7 (Num.)	71
8 (Num.)	72
9 (Num.)	73
4 (Num.)	75
5 (Num.)	76
6 (Num.)	77

1 (Num.)	79
2 (Num.)	80
3 (Num.)	81
0 (Num.)	82
Entfernen	83

Die Umsetzung des Scancodes in Zeichen erfolgt durch länderspezifische Tabellen, die Teil des Betriebssystems oder Teil von Anwendungen sind.

B Hardware-Interruptroutinen

In der Programmiersprache C werden solche Routinen durch das Voranstellen des Schlüsselworts "interrupt" vor dem eigentlichen Prozedurnamen vom Compiler selbständig erkannt. Mit den Funktionen "getvect (#of INT)" und "setvect (#of INT, Proc)" können die Einträge des entsprechenden Interrupts in der IDT gelesen bzw. überschrieben werden. Am Beispiel von Auszügen des Quellcodes der Antriebssteuerung kann das anschaulich gezeigt werden. Es soll hier eine Routine mit dem Namen "handler" jedesmal dann aufgerufen werden, wenn die IRQ-Leitung mit der Nummer 5 aktiviert wurde. Dabei ist zu beachten, daß vor dem Beenden des Hauptprogramms die ursprüngliche Interruptroutine (= Interrupthandler) wieder eingesetzt werden muß, damit das System auch weiter fehlerfrei läuft. Weitere Anmerkungen sind jeweils den Kommentaren am Zeilenende zu entnehmen.

```
#ifdef __cplusplus           //Die Interruptroutine soll unter C und
#define __CPPARGS ...       //C++ compilierbar sein
#else
    #define __CPPARGS
#endif

int INTR = 5;                //zu ersetzender Hardware-Interrupt
void interrupt handler(__CPPARGS);
void interrupt (*oldhandler) (__CPPARGS); // Prototyp, dem die alte
                                           // alte Interruptroutine zugewiesen wird

void interrupt handler(__CPPARGS) //neue Interruptroutine
{
    /*Hier können beliebige Anweisungen stehen,
    die KEINE Software-Interrupts auslösen */

    outp(0x20, 0x20);        //Rückstellen des Interruptcontrollers
}

void main()                  //Beginn des Hauptprogramms
{
    oldhandler = getvect(INTR); //Startadresse des alten Handlers wird
                                //ausgelesen und zugewiesen
    setvect(INTR, handler);    //neuer Handler wird eingesetzt

    /*Beliebige Anweisungen*/

    setvect(INTR, oldhandler); //alten Handler in die IDT eintragen
}
```

Folgende Punkte müssen bei der Programmierung von individuellen Hardware-Interrupt-routinen immer beachtet werden:

1. Der ursprünglich vorhandene Handler muß vor dem Programmende des Hauptprogramms wieder eingesetzt werden.
2. In einer Interruptroutine sollte keine Anweisung auftreten, die einen weiteren Interrupt aufruft. Dies führt üblicherweise zum Absturz des Systems. Andere Interrupts dürfen innerhalb einer Interruptroutine nur unter besonderen Vorsichtsmaßnahmen aufgerufen werden, die sehr komplex und in Hochsprachen nur schwierig zu realisieren sind.
3. Innerhalb der Interruptroutine muß der entsprechende Interruptcontroller wieder zurückgesetzt werden. Wurde z.B. eine Interrupt-Leitung aktiviert, die am kaskadierten, zweiten Controller eingeht, so müssen beide zurückgesetzt werden. Andernfalls bleibt das System stehen, da der Interruptcontroller immer noch die Bearbeitung eines Interrupts annimmt und so den Fortgang des Hauptprogramms blockiert.
4. Werden mehrere Interruptroutinen ersetzt, dann ist jede einzelne wie oben beschrieben zu behandeln.

C Zeitmeß-System mit geringem Meßfehler

Zur genauen Ermittlung der Laufzeiten von Prozeduren, ist das Auslesen der Systemuhr wenig geeignet. Messungen haben ergeben, daß allein die Standardfunktion in C (gettime) schon eine Laufzeit von ca. 25 µs benötigt. Eine genauere Möglichkeit ist die Zeitmessung mit einem externen Zeitwerk, das einen Interrupt anfordern kann.

Zunächst wird eine Interruptroutine für den vom Zeitwerk angeforderten Interrupt nach obigem Schema eingesetzt. Sie enthält außer dem Rücksetzbefehl für den Interruptcontroller nur das Setzen eines Flags.

```
int schluss = 0;
void interrupt cali_handler4( __CPPARGS)
{
    schluss=1;
    outp(0x20, 0x20);
}
```

Alle weiteren Interrupts werden über das Interrupt-Mask-Register ausgeschaltet und anschließend wird das Zeitwerk auf den höchstmöglichen Wert T_{max} im Single-Shot-Modus gesetzt (65355/ f_{Takt} bei Zeitwerken vom Typ 8254). Während das Zeitwerk arbeitet, kann in einer Schleife, deren Abbruchkriterium das gesetzte Flag der Interruptroutine ist, die zu messende Routine wiederholt ausgeführt werden. Die Anzahl der Routinenaufrufe wird mit der Inkrementierung eines Zählers n ermittelt.

```
void test()
{
    /*Beliebige Anweisungen*/
}

void main()
{
    while (!schluss)                //solange (schluss=0) do
    {
        test();
        n++;
    }
}
```

Nach Ablauf des geladenen Intervalls wird der Interrupt ausgelöst und das Abbruchkriterium gesetzt, dessen nächste Überprüfung den Ausprung aus der Schleife bedingt. Mit Hilfe der Schleifenzählvariablen n und dem in das Zeitwerk geladenen Intervalls T_{max} kann nun die Laufzeit t_{Lanf} der aufgerufenen Routine `test()` in guter Näherung mit

$$T_{Lanf} \approx \frac{T_{max}}{n}$$

berechnet werden.

Bei dieser Methode spielt der Zeitbedarf des Abbruch-Interrupts selbst keine Rolle, da der Programmfluß des Hauptprogramms angehalten ist. Die Ungenauigkeit der Methode ist kleiner als die einfache Laufzeit T_{Lauf} der Routine. Anzumerken ist, daß die Routine `test()` durchaus ebenfalls eine Interrupt-Service-Routine darstellen kann. Dies tritt z.B. bei der Messung der Interruptzeiten auf. `test()` ist dann als Interruptroutine zu deklarieren und darf nicht explizit aufgerufen werden. Der Aufruf erfolgt durch das System selbst, sobald der entsprechende Interrupt angefordert wird. Die Zählvariable n muß dann aber Teil der Interruptroutine sein.